MST210

Mathematical methods, models and modelling

Computer algebra guide

This publication forms part of an Open University module. Details of this and other Open University modules can be obtained from Student Recruitment, The Open University, PO Box 197, Milton Keynes MK7 6BJ, United Kingdom (tel. +44 (0)300 303 5303; email general-enquiries@open.ac.uk).

Alternatively, you may visit the Open University website at www.open.ac.uk where you can learn more about the wide range of modules and packs offered at all levels by The Open University.

The Open University, Walton Hall, Milton Keynes, MK7 6AA.

First published 2019.

Copyright © 2019 The Open University

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, transmitted or utilised in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without written permission from the publisher or a licence from the Copyright Licensing Agency Limited, 5th Floor, Shackleton House, Hay's Galleria, 4 Battle Bridge Lane, London SE1 2HX (website www.cla.co.uk).

Open University materials may also be made available in electronic formats for use by students of the University. All rights, including copyright and related rights and database rights, in electronic materials and their contents are owned by or licensed to The Open University, or otherwise used by The Open University as permitted by applicable law.

In using electronic materials and their contents you agree that your use will be solely for the purposes of following an Open University course of study or otherwise as licensed by The Open University or its assigns.

Except as permitted above you undertake not to copy, store in any medium (including electronic storage or use in a website), distribute, transmit or retransmit, broadcast, modify or show in public such electronic materials in whole or in part without the prior written consent of The Open University or in accordance with the Copyright, Designs and Patents Act 1988.

Edited, designed and typeset by The Open University, using LaTeX.

WEB 06698 3

1.1

Contents

Introduction **4**

1 Installing, testing and configuring Maxima **6**
1.1 Introducing and installing Maxima 6
1.2 Starting, testing and configuring wxMaxima 8

2 Getting started with Maxima **12**
2.1 Calculating with Maxima 12
2.2 Decimal numbers 16
2.3 Troubleshooting wxMaxima 17
2.4 Saving and printing your work 18
2.5 Getting help 20

3 Variables and functions **22**
3.1 Variables 22
3.2 Defining functions 25

4 Lists **27**
4.1 Using lists 27
4.2 Summing lists and series 30

5 Vectors **31**

6 Matrices **32**

7 Plotting graphs **35**
7.1 Plotting graphs of functions 36
7.2 Changing the appearance of a graph 38
7.3 Plotting graphs of discrete data 41

8 Solving equations **43**
8.1 Solving equations analytically 43
8.2 Solving equations numerically 44

9 Calculus **46**

10 Accessibility **50**
10.1 Choosing and configuring your interface 51
10.2 Starting, testing and configuring Maxima using the command-line interface 52
10.3 Troubleshooting Maxima using the command-line interface 55
10.4 Reusing and editing commands in the command-line interface 56
10.5 Saving and printing your work using the command-line interface 57
10.6 Plotting graphs without wxMaxima 60
10.7 Changing properties of graphs 60

Solutions to Computer activities **62**

Maxima reference guide **75**

Acknowledgements **80**

Index **81**

Introduction

Throughout your study of MST210, you will use the Maxima computer algebra system (CAS) to solve problems and perform calculations that may be too large or complicated to do by hand. Maxima is a powerful system that you may have met before on previous modules you have studied.

The use of Maxima is an important part of MST210 and its assessment. The aim of this guide is to ensure that you are able to use it as required. If you have used Maxima in a previous module, you may wish to skim through this guide and use it for reference. If you have not used Maxima before, it is recommended that you work through this guide and the activities it contains in detail.

Maxima can be used on many different types of computer through a variety of interfaces. This guide concentrates on using Maxima on a Microsoft Windows personal computer through the wxMaxima interface. This is the interface that you are recommended to use. All the Maxima commands mentioned in this guide can be used with any interface to Maxima, unless indicated otherwise.

Section 1 gives an overview of Maxima and includes activities in which you can install, test and configure Maxima on your computer. If you already have a working copy of Maxima ready to use, you can skip this section.

Section 2 describes how to perform calculations, and how to save and print your work.

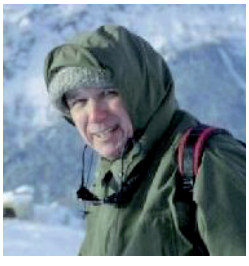
Sections 3 to 6 outline how to use variables, functions, lists, vectors and matrices.

Section 7 describes how to plot the various types of graphs you may need during MST210, Section 8 shows how Maxima can be used to solve equations and Section 9 explains how to find the derivative or integral of an expression.

Finally, Section 10 contains information on the accessibility of Maxima and is primarily aimed at those who may have difficulty using Maxima because of a disability. It provides some advice and suggestions of alternative ways of working to those given in the main text.

At the end of this guide is a summary of all the Maxima commands mentioned, which should be useful for reference.

Note that this guide does not attempt to describe all the features of Maxima. If you need more information, please consult the extensive documentation provided on the Maxima website at `maxima.sourceforge.net` or use Maxima's help system, which is described in Subsection 2.5.



William Schelter (1947–2001)

Maxima's origins lie in a system called Macsyma, a well-respected computer algebra system first developed in the late 1960s at the Massachusetts Institute of Technology and funded by the United States Department of Energy. From 1982 until his sudden death in 2001, William (Bill) Schelter, a professor of mathematics at The University of Texas at Austin, maintained a version of Macsyma which we now know as Maxima. In 1998 he obtained permission from the Department of Energy for the system to be made freely available to all. Maxima is now dedicated to his memory.

Conventions

Here are the various computing terms used in this guide, and how they should be interpreted on Microsoft Windows and Apple Mac computers. The guide concentrates on using Maxima on a Microsoft Windows computer, although Maxima is available for many different types of computer.

Computer terminology

Term	Microsoft Windows	Apple Mac
Click	Click the left mouse/touchpad button	Click the mouse/touchpad button
Right click	Click the right mouse/touchpad button	Hold the Ctrl key while clicking the mouse/touchpad button
Select	*Either* click to select an option, *or* select text by dragging the cursor over the text while pressing the left mouse button	*Either* click to select an option, *or* select text by dragging the cursor over the text while pressing the mouse button
`Ctrl-Q` Etc.	Hold down Ctrl while pressing Q Etc.	Hold down Command while pressing Q Etc. – but there are exceptions to this pattern that we'll mention as they arise

On some computer keyboards the Enter key might be labelled Return, or simply ↵, and the Shift key might be labelled ⇑.

In this guide, these keys are referred to as Enter and Shift, respectively.

Sometimes Enter and Return are two separate keys. For such keyboards references in this guide to the Enter key should be interpreted as references to the Return key.

1 Installing, testing and configuring Maxima

If you already have a working version of Maxima installed on your computer, for example from your study of a previous module, then you can continue to use that version and can skip this section.

1.1 Introducing and installing Maxima

Maxima can be used on many different types of computer, and there are several different intermediary systems, known as **interfaces** or **front-ends**, that provide a means of communication between you, the user, and the Maxima system itself. This is illustrated in Figure 1. The interfaces enable you to enter commands that are then processed by Maxima, and they display the results obtained.

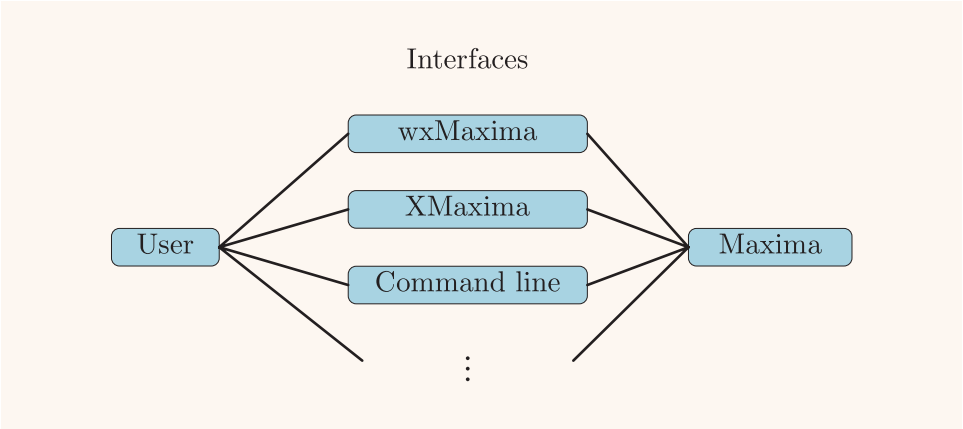


Figure 1 Different interfaces for Maxima

All the commands mentioned in this guide can be used with any interface to Maxima, unless indicated otherwise.

The recommended interface is **wxMaxima**, which is shown in Figure 2. (The letters 'wx' in the name of this interface refer to the software used to create it.) If you need to use an interface other than wxMaxima, for example if you are not using a Windows computer and you are unable to install wxMaxima, then you should try to achieve all the outcomes shown in this guide using your chosen interface.

If you use a screenreader, then the basic **command-line** interface shown in Figure 3 may be the easiest interface for you to use.

If you think you may have difficulty using the wxMaxima interface, because of a disability for example, then you should read Subsection 10.1 now. This contains details of how wxMaxima might be configured to meet your needs, and should help you decide whether to use wxMaxima or the command-line interface.

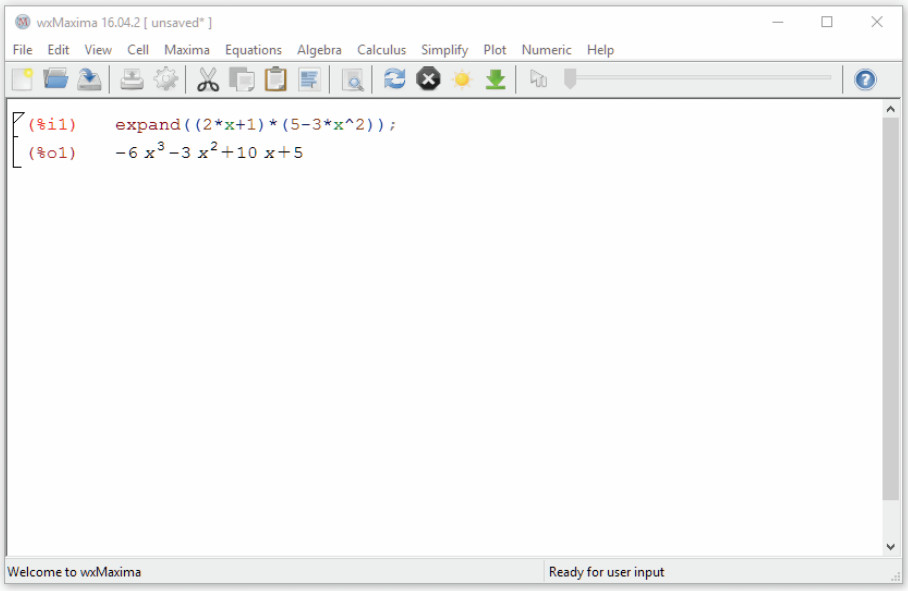

wxMaxima 16.04.2 [unsaved*]
File Edit View Cell Maxima Equations Algebra Calculus Simplify Plot Numeric Help
(%i1) expand((2*x+1)*(5-3*x^2));
(%o1) -6 x^3-3 x^2+10 x+5
Welcome to wxMaxima
Ready for user input

Figure 2 The wxMaxima interface

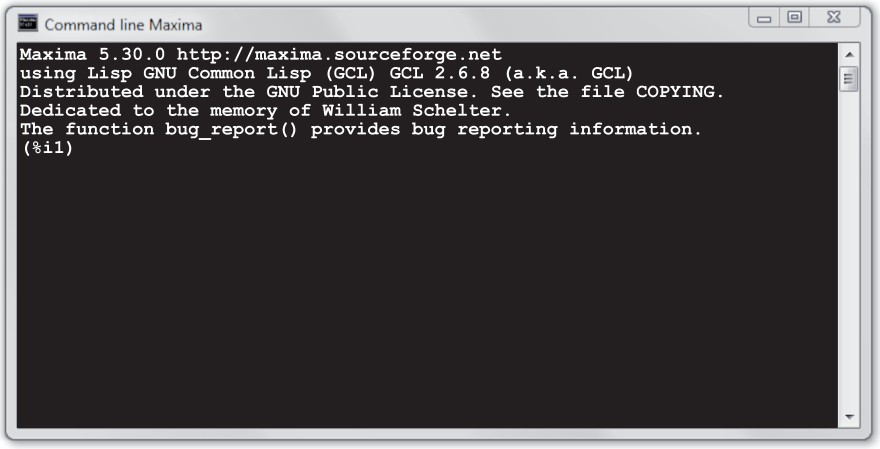

Command line Maxima
Maxima 5.30.0 http://maxima.sourceforge.net
using Lisp GNU Common Lisp (GCL) GCL 2.6.8 (a.k.a. GCL)
Distributed under the GNU Public License. See the file COPYING.
Dedicated to the memory of William Schelter.
The function bug_report() provides bug reporting information.
(%i1)

Figure 3 The command-line interface to Maxima

Before you can use Maxima, you need to install it on your computer. You will do this in the next activity.

Computer activity 1 Installing Maxima

(a) Go to the OU Maxima website at `learn1.open.ac.uk/site/maxima` and follow the instructions there to download and install Maxima on your computer. (You can also find a link to the OU Maxima website on the MST210 website.)

(b) Create a folder on your computer in which to store your Maxima work for this module. You might like to put this on your computer desktop, or in your usual documents area.

If you are studying within a prison or other closed institution, you can install Maxima from the *Offline resources* disc.

1.2 Starting, testing and configuring wxMaxima

If you are using the command-line interface you should study Subsection 10.2 instead of this subsection. If you are using a different interface, you should make sure that you can achieve similar outcomes with that interface.

Computer activity 2 Starting wxMaxima

Start wxMaxima on your computer, and keep it open to work with as you study this section.

If you cannot remember how to do this on your computer, see the OU Maxima website: `learn1.open.ac.uk/site/maxima`.

The wxMaxima interface to Maxima, illustrated in Figure 4, consists mainly of a large white working area (called the **worksheet**) where you can type Maxima commands. These commands instruct Maxima to either perform calculations or change system settings.

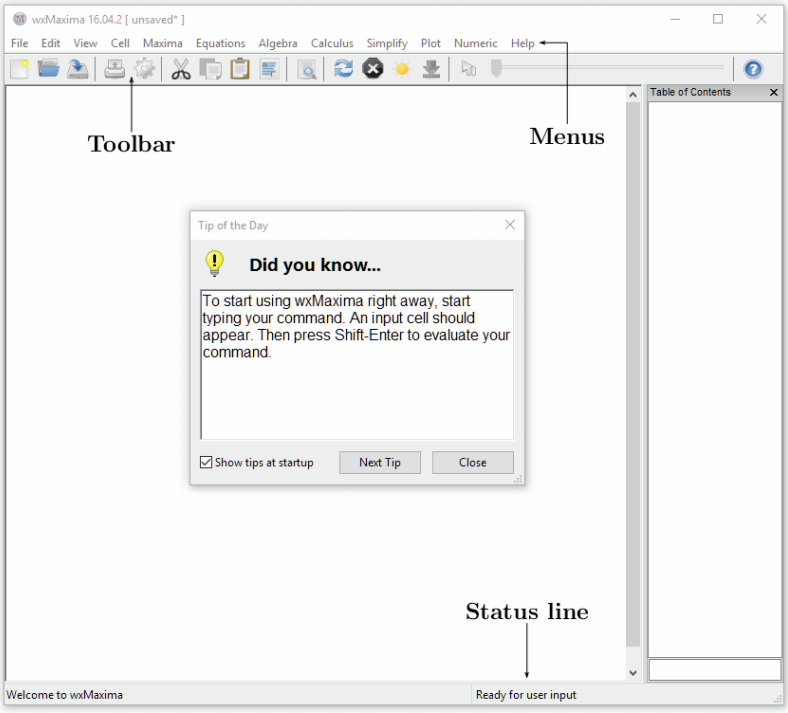

wxMaxima 16.04.2 [unsaved*]
File Edit View Cell Maxima Equations Algebra Calculus Simplify Plot Numeric Help
Table of Contents
Toolbar
Menus
Tip of the Day
Did you know...
To start using wxMaxima right away, start typing your command. An input cell should appear. Then press Shift-Enter to evaluate your command.
Show tips at startup
Next Tip
Close
Status line
Welcome to wxMaxima
Ready for user input

Figure 4 The initial wxMaxima window

When the software first starts, a secondary window showing the 'Tip of the Day' is also displayed. You can close it by clicking the 'Close' button. If you want to prevent this window from appearing the next time you start wxMaxima, click the box in its bottom left-hand corner, so that it is no longer ticked, before clicking 'Close'. (You can always reinstate these messages at any time by selecting `Show tips` from the `Help` menu.)

Along the top of the main window (or along the top of the screen in the case of an Apple Mac) is a list of menu names, including the `File` menu which allows you to save or print your work and re-open previously saved

sessions (there is more on this later). You can also close the software from the `File` menu by selecting `Exit`. (On an Apple Mac you can do this by selecting `Quit wxMaxima` from the `wxMaxima` menu.)

In Figure 4, a row of icons known as the **toolbar** is shown below the row of menu names. This is displayed only if `Main Toolbar` is selected in the `View` menu. The icons provide shortcuts to commonly used functions.

The `View` menu also allows you to display palettes of commonly used commands. You can use these palettes as an alternative to typing some of the commands given in this guide.

Also shown in Figure 4 is the **Table of Contents**. This shows the names of any sections or subsections that you may add to your worksheet. You may want to close the Table of Contents, by clicking the cross near its top right-hand corner.

At the bottom right-hand corner of the wxMaxima window is a message that indicates the current status of the Maxima system. When Maxima is waiting for instructions from you this reads 'Ready for user input'. Other possible messages include 'Maxima is calculating', or 'Reading Maxima output'.

The following activity shows you how to use wxMaxima to perform a simple calculation and checks that your system is working properly. Follow the steps using wxMaxima yourself as you read the activity.

If you encounter unexpected problems when working through this activity, check the Frequently asked questions (FAQs) section on the OU Maxima website: `learn1.open.ac.uk/site/maxima`.

Computer activity 3 Your first Maxima calculation

Follow these steps to use Maxima to calculate $2 + 3 \times \frac{4}{5}$.

(a) Click anywhere on the large white area of the wxMaxima window, to ensure that the text you type next is directed to the correct place.

(b) Type the following exactly as it appears here:

```
2+3*4/5;
```

including the semicolon (`;`) at the end.

Notice that as you type your command, it is shown in wxMaxima preceded by a red arrow: `-->`.

This is the **input prompt** and shows where commands can be typed.

To the left of the input prompt is the **cell marker**

which will be described in more detail later.

As you type, the colour of the text changes to highlight the different elements of your expression: numbers are coloured differently to the operations.

(c) Press `Shift-Enter`, that is, hold down the Shift key while pressing Enter.

(Remember: if, as on some Macs, your keyboard has separate Enter and Return keys, then you should hold down the Shift key while pressing Return.)

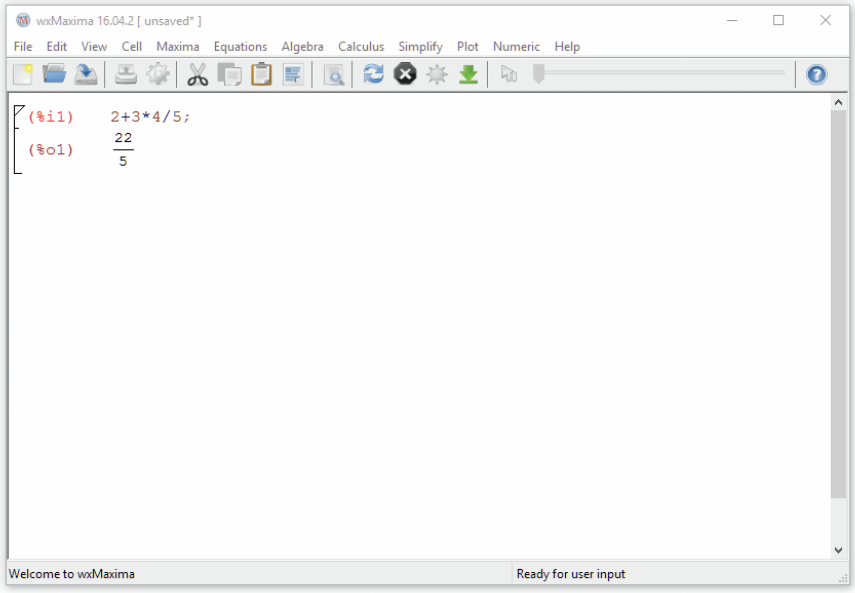
If you make a mistake, try clicking elsewhere on the worksheet, below your error, and try again.

Pressing `Shift-Enter` instructs wxMaxima to send the expression to Maxima to calculate. This is known as **evaluating** the expression.

The input prompt (`-->`) is now replaced by `(%i1)`. This is how Maxima labels lines. The `i` stands for 'input': this is input line number 1. Within each Maxima session line numbers start at 1 and increase for each new input line. The percentage symbol (`%`) is used by Maxima to indicate objects built into the system. There is more about this later.

The status message in the bottom right-hand corner of the wxMaxima window first changes to 'Maxima is calculating', then 'Reading Maxima output', before the result of the calculation is displayed. The message then displays 'Ready for user input' when the system is ready to accept another command.

The result is displayed under the input line, preceded by the label `(%o1)`. The `o` stands for 'output', so this label identifies the line as output line 1. It gives the output corresponding to input line 1.


wxMaxima 16.04.2 [unsaved*]
File Edit View Cell Maxima Equations Algebra Calculus Simplify Plot Numeric Help
(%i1) 2+3*4/5;
(%o1) 22/5
Welcome to wxMaxima
Ready for user input

If, in Computer activity 3, Maxima behaved as described, then you have a working system to use throughout MST210. Before finishing though, you should make a couple of small changes to how the wxMaxima interface behaves.

In Computer activity 3 you used `Shift-Enter` rather than Enter to instruct Maxima to evaluate expressions, and the input prompt (-->) appeared only once you started typing. Both these behaviours can be disconcerting, but can be changed, as described in Computer activity 4.

It is recommended that you configure wxMaxima as described in Computer activity 4, and in the remainder of this guide it is assumed that you have done so.

Computer activity 4 Configuring wxMaxima

(a) From the `Edit` menu, select `Configure` (or on an Apple Mac computer, from the `wxMaxima` menu, select `Preferences`).

Alternatively, click the following icon on the toolbar.

This opens the 'wxMaxima configuration' window.

(b) Within the Configuration window, click the icon labelled 'Worksheet' to access the worksheet settings, then tick the 'Enter evaluates cells' box by clicking it.

This changes the behaviour so that pressing Enter alone evaluates commands.

(c) In the same list of settings, tick the 'Open a cell when Maxima expects input' box by clicking it.

This ensures that the input prompt is shown whenever Maxima is waiting for input.

(d) In the same list of settings, ensure the 'Show user-defined labels instead of (%oxx)' box is *not* ticked.

This ensures that the output line numbers are always shown.

(e) Click on the icon labelled 'Options' in the Configuration window, then tick the 'Save sidebar status' box by clicking it.

This ensures that your preferred configuration of the wxMaxima window, for example hiding the Table of Contents, is retained between Maxima sessions.

(f) Click 'OK' (or close the window if using an Apple Mac).

Closing wxMaxima

When you have finished using Maxima, you can close Maxima and the wxMaxima interface by using one of the following methods.

- Select `Exit` from the `File` menu (or `Quit wxMaxima` from the `wxMaxima` menu on an Apple Mac computer).
- Press `Ctrl-Q`. (Here, `Q` stands for quit.)
- On a Windows computer, click the usual small cross button at the top right-hand corner of the wxMaxima window.

You may be asked if you want to save your changes to the worksheet before closing. At this stage, click 'No'. You will see how to save your work in Subsection 2.4.

2 Getting started with Maxima

In this section you will review how to perform calculations with Maxima. If you are familiar with Maxima from a previous module you have studied, you should be able to quickly skim through this section, perhaps trying some of the activities to check your skills.

If you encounter unexpected problems when working through this section, remember to check the Frequently asked questions (FAQs) section on the OU Maxima website.

2.1 Calculating with Maxima

The way in which a calculation or a command has to be entered in Maxima is known as its **syntax**. The Maxima syntax for the basic mathematical operations and other functions is summarised below. You can practise using some of these commands in Computer activity 5.

In this guide we will use ^ for powers. Many Maxima commands take the form of a command name, such as `sqrt` or `abs`, followed by a pair of round brackets containing one or more objects, such as numbers, that the command operates on. Such an object is called an **argument** of the command. In the summary tables in this guide, such as the one below, arguments of commands are often represented by ▯.

Like many computer systems, Maxima assumes that all angles are measured in radians. The names of the inverse trigonometric functions arcsin, arccos and arctan in Maxima are `asin`, `acos` and `atan` respectively. This notation is used by many computer systems.

Mathematical operations and functions

Operation or function	Syntax	Example
Addition	`+`	`4+3;`
Subtraction	`-`	`4-3;`
Multiplication	`*`	`4*3;`
Division	`/`	`4/3;`
Brackets	`(` *and* `)`	`2*(3+4);`
Powers, for example 2^3	`^`	`2^3;`
	or `**`	`2**3;`
Square root, for example $\sqrt{5}$	`sqrt(`␣`)`	`sqrt(5);`
Exponential, for example e^3	`%e^`␣	`%e^3;`
	or `exp(`␣`)`	`exp(3);`
Natural logarithm, ln	`log(`␣`)`	`log(8);`
Magnitude (absolute value)	`abs(`␣`)`	`abs(-3);`
sin	`sin(`␣`)`	`sin(1);`
cos	`cos(`␣`)`	`cos(3*%pi/2);`
tan	`tan(`␣`)`	`tan(%pi/4);`
cosec	`csc(`␣`)`	`csc(2);`
sec	`sec(`␣`)`	`sec(%pi/3);`
cot	`cot(`␣`)`	`cot(%pi/4);`
arcsin	`asin(`␣`)`	`asin(sqrt(3)/2);`
arccos	`acos(`␣`)`	`acos(1/sqrt(2));`
arctan	`atan(`␣`)`	`atan(1/2);`
sinh	`sinh(`␣`)`	`sinh(1);`
cosh	`cosh(`␣`)`	`cosh(1);`
tanh	`tanh(`␣`)`	`tanh(1);`
arcsinh	`asinh(`␣`)`	`asinh(1);`
arccosh	`acosh(`␣`)`	`acosh(1);`
arctanh	`atanh(`␣`)`	`atanh(1/2);`

The symbols `%e` and `%pi` in the table above are the Maxima representations of the mathematical constants e and π. The `%` symbol indicates that this is the name of a special quantity built into Maxima. When displaying mathematics in an output line, wxMaxima displays `%pi` as π. Another constant that you may see in the result of a Maxima calculation is `%i`, which represents i, the imaginary number whose square is -1.

Mathematical constants

Constant	Syntax	Example
e	`%e`	`%e^2;`
π	`%pi`	`2*%pi;`
i	`%i`	`2+3*%i;`

There is no Maxima command for logarithms to bases other than e. Logarithms to other bases can be found using:

$$\log_b x = \frac{\log_a x}{\log_a b}.$$

In particular, $\log_{10} x$ can be calculated in Maxima using

```
log(x)/log(10)
```

You should end all Maxima commands with either a semicolon (`;`) or a dollar sign (`$`). Ending a command with a dollar sign tells Maxima to perform the instruction, but not to display the answer. When using wxMaxima, if you forget to finish your command with a semicolon or dollar sign, then the system will add a semicolon for you. Other interfaces may not do this. In such systems, pressing Enter without ending the line with a semicolon or dollar sign simply moves the editing cursor to the next line, waiting for you to finish entering a complete command.

You can also enter several commands on one line, ending each one with a semicolon or dollar sign.

Maxima generally ignores all spaces that you type while entering a calculation or command, so you can use spaces to make a command easier to read. Once wxMaxima has been configured to use Enter to evaluate commands (as described in Computer activity 4), you can then use `Shift-Enter` to enter line breaks, which are also ignored by Maxima.

Warning: Do not omit multiplication signs!

A common error when first using Maxima is to omit multiplication signs. For example, you might type

- `2sqrt(2)` rather than `2*sqrt(2)`, or
- `2(3+4)` rather than `2*(3+4)`.

Doing this will result in an error. The displayed error message might include one of the following statements:

- *parser: incomplete number; missing exponent?*
- *incorrect syntax: Syntax error*
  incorrect syntax: Too many)'s
  incorrect syntax: Premature termination of input at ;.
- *... is not an infix operator.*

If you receive one of these error messages, first check for missing multiplication signs!

Warning: Take care with question marks (?)

Do not type question marks when asking Maxima to evaluate an expression. The symbol `?` is a special symbol in Maxima which provides access to underlying systems.

The one exception to this is when you are using the commands for obtaining help from the system, which you will see in Subsection 2.5.

You can practise using Maxima for mathematical calculations in the following activity.

Computer activity 5 Calculating with Maxima

Calculate the following using Maxima.

(a) $\sqrt[5]{32}$

Since $\sqrt[5]{32} = 32^{\frac{1}{5}}$ this can be found by entering

```
32^(1/5);
```

(The brackets in this expression are required since Maxima calculates expressions using the correct order of mathematical precedence.)

Remember to press Enter to evaluate the command.

(b) $\sin\left(\frac{\pi}{3}\right)$

Remember that to enter the constant π in Maxima, you type `%pi`.

Maxima returns an exact answer, in the form of a surd.

(c) $\tan 30°$

Remember to convert the angle to radians, by multiplying 30 by $\frac{\pi}{180}$.

(d) e^4

Remember, the Maxima symbol for e is `%e`, or you can use the `exp(  )` function.

Since there is no simpler exact way of expressing e^4, Maxima leaves this expression unevaluated.

(e) $\ln 3.4$

Remember that the Maxima command for a natural logarithm is `log(  )`. Maxima has no command `ln(  )`.

Here, since the argument is a decimal number, Maxima returns a decimal approximation to the answer.

During the previous activity you may have noticed that when you enter an opening bracket, the wxMaxima interface automatically adds a closing one for you. This behaviour can be turned off by unticking 'Match parenthesis in text controls' in the 'Worksheet' settings of the Configuration window. (See Computer activity 4 for how to open the Configuration window.)

2.2 Decimal numbers

In Computer activity 5 you saw that Maxima usually tries to calculate quantities *exactly* rather than approximating them by decimal numbers. It does this by manipulating the symbols in the expression according to the rules of mathematics. This is sometimes known as **symbolic computation**. If an exact answer cannot be obtained, the expression will be returned unevaluated, as in Computer activity 5(d). The exception to this is if the expression to calculate contains a decimal number, such as in Computer activity 5(e), when Maxima will work in terms of decimal numbers.

If you wish to have a result displayed as a decimal number, you can instruct Maxima to do so with the `float` command. The name of the command `float` arises from the fact that the method used by computers to store decimal numbers internally using the binary digits 0 and 1 is called a *floating-point representation*. The `float` command instructs Maxima to convert a number stored symbolically as a mathematical expression to one stored using a floating-point representation.

Alternatively, the text `,numer` can be added after a command.

Computer activity 6 Obtaining exact and approximate answers

In Maxima, calculate 1309/3451 in three different ways, by entering each of the following expressions in turn.

(a) `1309/3451;` (b) `1309.0/3451;` (c) `float(1309/3451);`

Maxima returns an exact answer for part (a).

Since part (b) involves a decimal number, 1309.0, Maxima returns a decimal approximation to the answer.

In part (c), the command `float` instructs Maxima to return a decimal answer.

Converting to decimal numbers

Operation	Command	Example
Convert to a decimal number	`float(  )` *or* `  ,numer`	`float(sqrt(2));` `sqrt(2),numer;`

Maxima usually displays decimal numbers to 16 significant figures (though it suppresses trailing zeros at the end of the decimal part of a number; for example 0.125 is displayed simply as `0.125` rather than as `0.1250000000000000`). This is also the precision to which Maxima performs decimal calculations.

You may also see decimal numbers displayed in the form

$$6.223015277861141\ 10^{139}$$

which is wxMaxima's way of displaying scientific notation. It means $6.223\,015\,277\,861\,141 \times 10^{139}$. (The command-line interface would display this as `6.223015277861141E+139`.) You can also enter numbers given in scientific notation. For example, to enter 6.022×10^{23}, you can type

```
6.022e23
```

(Note that the use of `e` here, without multiplication or percentage symbols, is different to the use of `%e`, which represents the mathematical constant e.)

Computer activity 7 Finding numerical approximations

(a) Find a decimal approximation of $\sqrt{325}$ using the `float` command.

(b) Find a decimal approximation of $\ln 10$ using the `numer` command.

2.3 Troubleshooting wxMaxima

If you are using wxMaxima and it does not seem to be responding, try the following suggestions.

(Suggestions for troubleshooting Maxima when using the command-line interface are given in Subsection 10.3.)

Troubleshooting wxMaxima

Cell Marker	Status message	Comments
Boxed:	Maxima is calculating	Maxima is evaluating a command that is taking a long time to complete.
Boxed:	Ready for user input	Maxima is waiting for you to type something before continuing. In either case you might wish to abort the command in one of the following ways. • Click on the toolbar. • Select `Interrupt` from the `Maxima` menu. • Type `Ctrl-G` (on an Apple Mac hold down the Command key while pressing .). Maxima might take some time to respond to this request.
	Maxima process terminated	Maxima (but not the wxMaxima interface) has stopped working. Select `Restart Maxima` from the `Maxima` menu.

If you encounter other problems, check the Frequently asked questions (FAQs) section on the OU Maxima website: `learn1.open.ac.uk/site/maxima`.

2.4 Saving and printing your work

(Details on saving and printing your work when using the command-line interface are given in Subsection 10.5.)

After working with Maxima you will probably want to save your calculations so that you can reuse or read them at a later date. You can do this in wxMaxima using the `Save` or `Save As` options from the `File` menu.

There are various different formats in which you can save your work.

It is recommended that you save your wxMaxima work as `The input without images` with the `.wxm` file type. This saves all your input but not any calculated output. You have to recalculate the output when you re-open the file, as demonstrated in Computer activity 8.

The alternative file type available is the `Whole document` with the `.wxmx` file type. This saves all the input and the output. Note that when you load a worksheet previously saved as a `.wxmx` file, although both the saved input and output are displayed, the input has not been evaluated by Maxima, so you cannot immediately use any results of the worksheet in new calculations. To re-evaluate the worksheet, and so recalculate all the results it contains, select `Evaluate All Visible Cells` from the `Cell` menu, or press `Ctrl-R`.

Input commands can also be exported to a `Maxima batch file`, with the `.mac` file type, using the `Export` option of the `File` menu. This is for more experienced users of Maxima.

You can open a saved wxMaxima worksheet by using the `Open` option from the `File` menu, or by double-clicking on the saved file on your computer.

Computer activity 8 Saving your work

Save your current worksheet then reload it, by following these steps.

(a) Select `Save As` from the `File` menu.

(b) In the window that appears:

(i) Choose the folder in which to save your work. You might like to save it in the folder that you created in Computer activity 1.

(ii) Choose a name for the file in which the worksheet will be saved.

(iii) Make sure the file is saved as a `The input without images` with the `.wxm` file extension.

(On a Windows computer, make sure that `The input without images (*.wxm)` is selected from the 'Save as type' drop-down list.)

(iv) Click 'Save'.

(c) Close wxMaxima, by selecting `Exit` from the `File` menu, or pressing `Ctrl-Q`, or clicking the small '`x`' at the top right of the window. (On a Mac select `Quit wxMaxima` from the `wxMaxima` menu or hold down the Command key while pressing Q .)

(d) Open your worksheet in wxMaxima again, by either

(i) double-clicking on the file you just saved, or

(ii) following the steps below:

- Start wxMaxima again.
- Select `Open` from the `File` menu. Maxima will ask if you want to save your just opened, blank worksheet. Click 'No'.
- Find the file containing your work and click 'Open'.

(e) Re-evaluate the worksheet by selecting `Evaluate All Visible Cells` from the `Cell` menu or pressing `Ctrl-R`.

The calculations in the worksheet are evaluated and results displayed.

Printing is best achieved by first exporting the whole worksheet to a HTML document (a web page) by selecting `Export` from the `File` menu, then opening the HTML document using a web browser and printing from that. The worksheet can also be exported as source code for the LaTeX mathematical typesetting system using the same menu items. (Currently, these exports are not perfect representations of the worksheet.)

Alternatively, you can obtain an image of the entire wxMaxima window by taking a 'screenshot'. To do this on a Microsoft Windows computer, first click on the wxMaxima window, then press `Alt-PrintScreen` (that is, hold down the Alt keyboard key while pressing the `PrintScreen` key, which may be labelled PrtScr, Prt Scrn or something similar). This stores the image, which you can then paste into a suitable application. If you are using Windows Vista, Windows 7, 8 or 10, you might like to use the 'Snipping Tool', available by typing 'snip' into the Start menu. To take a screenshot on an Apple Mac computer, press `Cmd-Ctrl-Shift-4` (that is, hold down the Command, Ctrl and ⇧ keyboard keys while pressing the 4 key). Next press the space bar, move the camera pointer over the wxMaxima window and click. This stores the image, which you can then paste into a suitable application. You may also like to use the Grab application from the Utilities folder.

You can also paste individual output lines (or parts of output lines) from the worksheet into, for example, word-processed documents. Select the part of the output that you wish to paste, then from the menu obtained by clicking the right mouse button, select one of the following options.

- `Copy` – This copies the Maxima syntax for the selection. It can be useful if you wish to use the Maxima syntax in another document, or use the copied output as input to another Maxima command.
- `Copy as LaTeX` – This copies a LaTeX representation of the selection that can be pasted into a source file of the LaTeX mathematical typesetting system.
- `Copy as MathML` – This copies the selection as MathML, which is an international standard for the representation of mathematics. The MathML can be pasted into, for example, versions of Word from 2007 onwards, where it will be pasted as mathematics that can be edited using the ribbon maths editor. It can also be imported into LibreOffice

documents, by, within LibreOffice, first creating a maths object using the `Insert > Object > Formula` menu options, then importing using `Tools > Import MathML from Clipboard`.

- `Copy as Image` – This copies the selection as an image that can be pasted into another document.

Whole cells can also be cut and pasted using the above options, by selecting the cell marker before clicking the right mouse button. However, when doing this, the LaTeX source contains a significant amount of formatting information that you may not require, and the MathML option for a whole cell does not work in the version of wxMaxima current at the time of writing.

2.5 Getting help

There are several ways in which you can obtain help with using Maxima, beyond this guide. In wxMaxima, you can access the complete Maxima manual by selecting `Maxima Help` from the `Help` menu. This opens the window shown in Figure 5. You can use the Contents, Index and Search tabs to look for the information that you want.

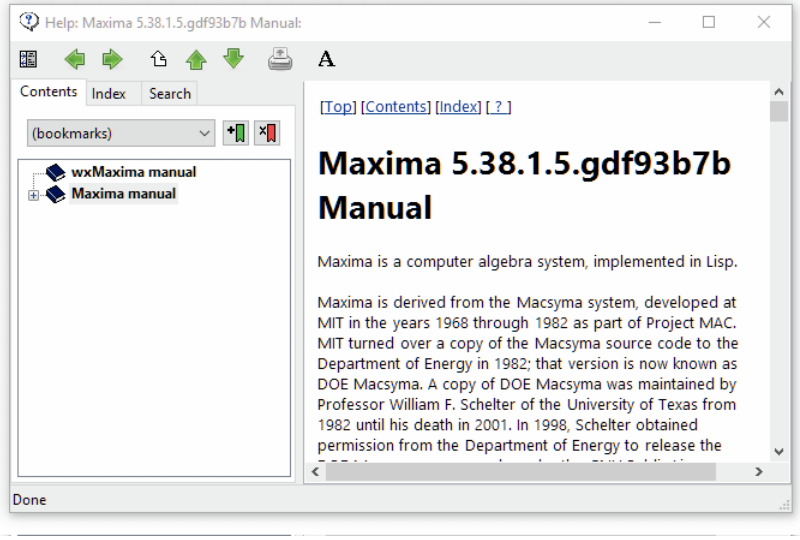

Help: Maxima 5.38.1.5.gdf93b7b Manual:
Contents
Index
Search
(bookmarks)
wxMaxima manual
Maxima manual
[Top] [Contents] [Index] [?]
Maxima 5.38.1.5.gdf93b7b Manual
Maxima is a computer algebra system, implemented in Lisp.
Maxima is derived from the Macsyma system, developed at MIT in the years 1968 through 1982 as part of Project MAC. MIT turned over a copy of the Macsyma source code to the Department of Energy in 1982; that version is now known as DOE Macsyma. A copy of DOE Macsyma was maintained by Professor William F. Schelter of the University of Texas from 1982 until his death in 2001. In 1998, Schelter obtained permission from the Department of Energy to release the
Done

Figure 5 The wxMaxima help system

Alternatively, Maxima also includes commands that you can use to obtain help. These are demonstrated in the following activity and summarised in the table that follows.

Computer activity 9 Getting help!

(a) Enter

```
? float;
```

to show the help for the `float` command.

Note the space between the `?` and `float`, which is needed. This is one of the few times in Maxima when spaces matter!

The '`?`' is the Maxima syntax to request help on the command given.

The help information for the command may include more detail than you need!

After giving the help information, Maxima displays the output `true`. This means that the command you entered was successful.

(b) Suppose that you cannot remember the Maxima command for square root, but you do remember that it was something like *sqr*.

Enter

```
?? sqr;
```

to list all the help information titles that contain the letter sequence 'sqr'.

A list of possible titles is displayed.

Title number 1 is what you were looking for, so enter `1;` to display the relevant information.

If more than one title is displayed in response to a `??` command, then Maxima will not continue until you enter one of the following:

- the appropriate number, to display your chosen information
- several numbers, separated by spaces, to see help information on several topics
- `all` to display the information on all topics listed, or
- `none` to see no information.

A box will be shown around the cell marker while Maxima is waiting for your response.

Help commands

Operation	Command	Example
Get help on a command	`? command` *or* `describe(command)`	`? float;` `describe(float);`
Find help information whose title contains the given text	`??` *or* `describe(, inexact)`	`?? flo;` `describe(flo, inexact);`

Note 1: the space after the `?` is needed.

Note 2: the second pair of commands displays a list of the titles of all help information pages whose title contains the given text. To obtain help on a particular topic listed, type the number corresponding to the required title in the list, followed by a semicolon.

There is further help available in the Frequently asked questions (FAQs) section of the OU Maxima website: `learn1.open.ac.uk/site/maxima`.

3 Variables and functions

In this section you will review how to assign numerical values and algebraic expressions to variables, and how to define your own functions.

3.1 Variables

Maxima allows you to assign a value or algebraic expression to a *variable*, and then use that variable in further calculations. Algebraic expressions are entered into Maxima using the same syntax used for numerical expressions described in Subsection 2.1.

For example, you can assign the value 23 to the variable `a`, and then use `a` in subsequent expressions or commands, such as $\mathtt{a}^2 - 3\mathtt{a} + 2$.

To assign a value or expression to a variable you use a colon (`:`). For example, the command

```
a:23;
```

assigns the value 23 to the variable `a`. The colon can be read as 'is assigned the value'.

Warning

You cannot use `=` to assign a value to a variable. This symbol is used to define *equations* in Maxima.

Variable names can be any combination of letters and numbers that begins with a letter. For example, Maxima will accept any of the following variable names.

```
a    A    solution    solution2    x2b
```

Some words are built-in to Maxima and cannot be used as variable names. If you try to assign such a variable, you will get an error such as:

```
assignment:  cannot assign to ...
-- an error.  To debug this try:  debugmode(true);
```

If you encounter such an error, choose a different name for your variable.

Variable names are *case-sensitive*; for example, the variable `x` is different from the variable `X`. If a variable name is the name of a Greek letter, then wxMaxima will display it in output lines as the proper Greek character; for example, the variable `alpha` will be displayed as α.

Once a variable has been assigned a value, Maxima will remember this value for the rest of your session. This may sometimes surprise you. It is easy to forget that earlier in the day you assigned a value to, say, `x`; you may then be confused later on when a calculation involving `x` gives an unexpected result. You can, however, tell Maxima to 'forget' about a variable to which you have previously assigned a value. You can also ask it to list all the variables that you have assigned values to. These commands are demonstrated in the following activity.

Computer activity 10 Working with variables

(a) Assign the expression $x^2 + 3$ to the variable `a` by entering

```
a:x^2+3;
```

The value of `a` is displayed as output.

Note that when assigning variables, wxMaxima precedes the output with either the output line number or the name of the variable being assigned a value. Which is shown is determined by whether or not 'Show user-defined labels instead of (%oxx)' is ticked in the 'Worksheet' settings of the Configuration window. In this guide, the output line numbers are shown.

(b) Enter `a;`

This displays the current value of `a`.

The value of the variable `a` can also be displayed using the command

```
print(a);
```

which displays the value of `a` on a new line, and then also in the corresponding output line.

(c) Assign $\texttt{a}^2$ to `b`.

(d) Display the values of `a` and `b` using the command

```
print(a,b);
```

The `print` command allows the value of more than one variable to be displayed on one line. Note that the values of the variables listed are displayed on a new line, next to each other (which might be misleading), and the value of the last variable given in the command is also shown on the corresponding output line.

(e) Edit the line where you assigned $x^2 + 3$ to `a`, so that the expression $x^3 + y$ is assigned to `a`.

Notice that terms are not always displayed in the order you expect!

(f) Enter `b;` to display the value of `b`.

Notice that the value of `b` has not changed. The variable `b` was defined when `a` had its original value.

(g) Enter `values;` to list the names of all the variables that are currently assigned values.

The variable names are given within square brackets, separated by commas. This is how Maxima displays *lists*. There is more about lists in Section 4.

(h) Enter `kill(b);` to remove the variable `b` from the system.

You should obtain the output *done*.

(i) Enter `b;` to display the value of `b`.

The variable name is output, as `b` no longer has a value.

(j) Enter `a;` to display the value of `a`.

`a` is still defined.

(k) Find the value of `a` when $x = 3$ by entering

```
subst(3,x,a);
```

The command name `subst` is short for 'substitute'.

The command `subst(3,x,a);` substitutes the value 3 for the variable `x` in the expression stored in the variable `a`.

(l) Find the value of `a` when $x = 2$ by entering

```
ev(a,x=2);
```

The command `ev` is short for 'evaluate'. This is an alternative way of substituting values into an expression.

The command `ev(a,x=2);` evaluates `a` when `x=2`.

(m) Find the value of `a` when $x = 2$ and $y = 1$ by entering

```
ev(a,[x=2,y=1]);
```

The `ev` command allows values of more than one variable to be given. The values are given as the second argument, within square brackets and separated by commas. This is another Maxima list. Lists are described in more detail in Section 4.

Notice that in Computer activity 10(e) and (f), when the value of `a` was changed, the value of `b` was not affected, since it was defined when `a` had its original value. If you want to update the value of `b` using a different value of `a`, then you should enter a new value of `a` then re-evaluate the line defining `b`. You can do this in wxMaxima by selecting the line and pressing Enter, or re-evaluating the whole worksheet by selecting the option `Evaluate All Visible Cells` from the `Cell` menu.

Working with variables

Operation	Command	Example
Assign a value to a variable	`variable:value`	`a:23;`
Display the value of a variable	`variable` `print(variable)`	`a;` `print(a);`
Display the value of several variables	`print(variable,variable,...)`	`print(a,b);`
List all user-assigned variables	`values`	`values;`
Remove an assigned variable	`kill(variable)`	`kill(a);`
Remove all assigned variables	`kill(all)`	`kill(all);`

Note: here the placeholder `variable` represents any variable name.

Substituting into an expression

Operation	Command	Example
Substitute	`subst(value,variable,expression)`	`subst(4,x,x^2+1);` which substitutes 4 for `x` in `x^2+1`
	`ev(expression,variable=value)`	`ev(x^2+1,x=4);` which substitutes 4 for `x` in `x^2+1`
	`ev(expression,` `[variable=value,variable=value,...])`	`ev(x^2+y,[x=1,y=2]);` which substitutes 1 for `x` and 2 for `y` in `x^2+y`

Note: `expression` indicates that the argument of the command must be an expression, or a variable whose value is an expression.

3.2 Defining functions

You can define your own functions in Maxima using the `:=` operator. This is demonstrated in the following activity. Like variables, function names can be any combination of letters and numbers that begins with a letter, and they are case-sensitive.

Computer activity 11 Defining functions

(a) Define the function f in Maxima, with the rule $f(x) = \dfrac{x^4 + x + 1}{x^3 - 13x + 12}$, by entering the following command. (Use the characters `:` followed by `=` to type the `:=` operator.)

```
f(x):=(x^4+x+1)/(x^3-13*x+12);
```

(b) Calculate the value of $f(2)$, by entering `f(2);`

(c) Try to calculate the value of $f(3)$.

This gives an error.

This is because when Maxima tries to evaluate $f(3)$, it finds that this involves dividing by zero, since the denominator of f is zero when $x = 3$, so it returns an error message. The function f is undefined at $x = 3$.

(d) Enter `g(1);`

> Maxima returns the expression `g(1)`, unevaluated.
>
> Since the function `g` has not been defined, Maxima cannot calculate a value. It knows only that the value is `g(1)`.

(e) Enter `functions;`

> This displays the value of the system variable `functions`, which is a list of all your currently defined functions in this Maxima session.

Notice the difference between defining a *function* using `:=` and assigning an expression to a *variable* using `:`. A function has an argument and can be evaluated at different values of the argument, while a variable does not. The variables `f` and `F`, for example, might appear in the same Maxima session as the functions `f(x)` and `F(x)`, but all four are different!

Once you have defined a function in Maxima it is remembered by the system for the rest of your session, unless you remove it using the `kill` command that you reviewed for variables in Subsection 3.1.

Functions can have more than one independent variable. Such functions are defined and used in the natural way, as demonstrated in the following activity.

Computer activity 12 Using a function of two variables

(a) Define the function $h(x, y)$ such that $h(x, y) = xy^2$ by entering

```
h(x,y):=x*y^2;
```

(b) Find $h(2, 3)$ by entering

```
h(2,3);
```

Working with functions

Operation	Command	Example
Define a function	`:=`	`f(x):=2*x+3;` `f(x,y):=x*y;`
Evaluate a function at a value	`function( )`	`f(1);`
List all user-defined functions	`functions`	`functions;`
Remove a defined function	`kill(function)`	`kill(f);`

4 Lists

In Maxima, a **list** is a collection of individual **elements** separated by commas and enclosed within square brackets. Lists are a powerful tool for storing and manipulating data.

4.1 Using lists

The use of lists is demonstrated in the following activity.

Computer activity 13 Working with lists

(a) Assign the list containing the first five prime numbers to the variable `A` by entering

```
A: [2,3,5,7,11];
```

(b) Display the fourth element of the list by entering

```
A[4];
```

> The position number of an element in a list is known as its **index**.
>
> You can obtain individual elements of a list by typing the index of the element in square brackets after the list, or after the name of a variable to which the list is assigned.
>
> Alternatively, each of the first 10 elements of a list can be obtained using the commands `first(  )`, `second(  )`, ..., `tenth(  )`, and the last element obtained using `last(  )`, where `  ` represents the name of the list.

(c) Use the command `third(  )` to obtain the third element of the list `A`.

(d) Use the command `last(  )` to obtain the last element of the list `A`.

(e) Enter

```
length(A);
```

to count the number of elements in the list `A`.

(f) Create a list `B` containing the first 7 prime numbers by extending the list `A` using the command

```
B: append(A, [13,17]);
```

and check the length of `B` is as expected.

> The `append` command creates a list by amalgamating the lists given as its arguments, in the order they are given.

The operations introduced in Computer activity 13 are summarised below.

Lists

Operation	Command	Example
Form a list	`[ , ,...]`	`[a,b,c];`
Assign a list to a variable	`variable:[ , ,...]`	`A:[a,b,c];`
Extract an element of a list	`list[index]`	`A[2];`
Extract the first, second, ..., tenth, element of a list	`first(list)`, `second(list)`, ..., `tenth(list)`	`first(A);`
Extract the last element of a list	`last(list)`	`last(A);`
Find the length of a list	`length(list)`	`length(A);`
Combine lists	`append(list,list,...)`	`append([1,2],[3,4]);`

When creating lists, you do not always have to type all the elements yourself: you can often get Maxima to create the list for you using the `makelist` and `create_list` commands. Some of the various ways these commands can be used are demonstrated in the following activity.

Computer activity 14 Creating more lists

(a) Create a list of the squares of the first 10 natural numbers and assign the list to the variable `a` by entering

```
a:makelist(i^2, i, 1, 10);
```

The `makelist` command creates a list. The first argument of the command is the general form of an element of the list. In this case, each element has the form i^2. This is given in terms of a dummy variable, which here is `i`.

The second argument of the `makelist` command is the name of the dummy variable used, and the third and fourth arguments are the minimum and maximum values taken by the dummy variable.

(b) Create a list of the cubes of the first 10 odd numbers by entering

```
makelist(i^3, i, 1, 19, 2);
```

Here, an additional final argument has been included in the command: this is the step by which the dummy variable is increased to create each element in the list. In this case, `i` starts at 1 and increases in steps of 2 to 19, giving the first 10 odd numbers.

(c) Create a list whose elements are those of the list `a` from part (a), each increased by 1, by entering

```
makelist(i+1, i, a);
```

Here, the third argument of the makelist command is a list (assigned to the variable `a`). The dummy variable `i` takes each value of the list in turn, and the elements of the resulting list are formed from `i+1`.

(d) Create a list whose elements are given by

$2^i 3^j$ where $i = 1, 2, 3$ and $j = 2, 4, 6$

by entering

```
create_list(2^i*3^j, i, [1,2,3], j, [2,4,6]);
```

Here, the general form of an element of the list contains two dummy variables: i and j. The `create_list` command allows lists to be created from expressions containing two or more dummy variables. The first argument is the general form of an element of the list. This is followed by the name of the first dummy variable and a list containing the values that the variable should take. This is followed by similar information for any other dummy variables.

(e) Create a list whose elements are given by

$x^i y^j$ where $i = 1, 2, 3$ and $j = 1, 2$

by entering

```
create_list(x^i*y^j, i,1,3, j,1,2);
```

This example uses an alternative form of the `create_list` command in which the minimum and maximum values taken by each dummy variable are given, rather than a list of possible values.

The `makelist` and `create_list` commands used in the previous activity are summarised below.

Creating lists

Operation	Command	Example

Create a list	`makelist(general term, dummy variable, start value, end value)`
	`makelist(general term, dummy variable, start value, end value, step)`
	`makelist(general term, dummy variable, list)`
	
	`create_list(general term, dummy variable 1, list, dummy variable 2, list, ...)`
	`create_list(general term, dummy variable 1, start value, end value, dummy variable 2, start value, end value, ...)`
		`makelist(2*n,n,1,100);`
		
		`makelist(2*n,n,1,100,2);`
		
		`makelist(2*n,n,A);`
		
		`create_list(x^i*y^j, i,[1,2,3], j,[2,4,6]);`
		
		`create_list(x^i*y^j, i,1,3, j,1,2);`

Note that the individual elements of a list can themselves be a list. For example, the list

```
[ [1,1], [2,4], [3,9] ]
```

is a list containing three elements, each of which is a list containing an integer and its square. You can practise working with a list of lists in the following activity.

Computer activity 15 Working with a list of lists

(a) Create a list whose elements are themselves lists containing an integer and its square, and assign the list to the variable `b` by entering

```
b:makelist([i,i^2], i, 1, 9);
```

(b) Display the sixth element of the list `b`.

As expected, this is itself a list.

(c) Display the second element of the list which is the sixth element of `b` by entering

```
b[6][2];
```

Here, `b[6]` is the sixth element of `b` which is `[6,36]`. So `b[6][2]` is the second element of `[6,36]` which is `36`.

4.2 Summing lists and series

The elements of a list can be summed using the `sum` command which is outlined below and demonstrated in Computer activity 16. The command can also be used to sum a mathematical series.

Summing lists and series

Operation	Command	Example
Sum the elements of a list or terms of a series	`sum(general term,dummy variable, start value,end value)`	`sum(i^2,i,1,100);`

Computer activity 16 Summing lists and series

(a) Sum the first four terms of the list `A` which was defined in Computer activity 13 using

```
A: [2,3,5,7,11];
```

by entering

```
sum(A[n],n,1,4);
```

The first argument of the `sum` command is the general form of the terms to be summed, given in terms of a dummy variable. Here the terms are the elements `A[n]` of the list `A`. The second argument is the name of the dummy variable, in this case `n`. The third and fourth arguments are the minimum and maximum values taken by the dummy variable. In this example these are `1` and `4`.

(b) Calculate $\sum_{i=1}^{100} i^2$ by entering

```
sum(i^2,i,1,100);
```

5 Vectors

In Maxima, lists can be used to represent row vectors, and vectors of equal length can be combined using the vector operations listed below. A column vector can be obtained by taking the transpose of a row vector.

Vector operations

Operation	Syntax	Example
Vector addition	`+`	`v+w;`
Vector subtraction	`-`	`v-w;`
Scalar multiple of a vector	`*`	`2*v;`
Vector scalar (dot) product	`.`	`v.w;`
Transpose	`transpose(vector)`	`transpose(v);`

The use of vectors is demonstrated in the following activity.

Computer activity 17 Using vectors

(a) Assign the column vector $\mathbf{v} = \begin{bmatrix} 2 \\ 3 \\ 6 \end{bmatrix}$ to the variable `v` by entering

```
v:transpose([2,3,6]);
```

(b) Assign the column vector $\mathbf{w} = \begin{bmatrix} 1 \\ -2 \\ 4 \end{bmatrix}$ to the variable `w` similarly.

(c) Find $2\mathbf{v}$.

> This is a scalar multiple of a vector, so use the `*` operator.

(d) Find $3\mathbf{v} + 2\mathbf{w}$.

(e) Find the scalar (or dot) product $\mathbf{v} \cdot \mathbf{w}$.

> The scalar (or dot) product of two vectors is found using the `.` operator.

(f) Find the magnitude of $\mathbf{v}$ by entering

```
sqrt(v.v);
```

> Since $\mathbf{v} \cdot \mathbf{v} = |\mathbf{v}|^2$, then $|\mathbf{v}| = \sqrt{\mathbf{v} \cdot \mathbf{v}}$.

Warning

Maxima does not strongly distinguish between row and column vectors, and so it is possible to use Maxima to add a row vector to a column vector. However, this is not good practice!

6 Matrices

There are several ways to input matrices in Maxima. Two of these are demonstrated in the following activity.

Computer activity 18 Entering matrices

(a) Assign the matrix $\begin{bmatrix} 2 & -3 & 3 \\ -2 & 1 & 2 \\ 1 & -1 & 4 \end{bmatrix}$ to the variable `P` by entering

```
P:matrix( [2,-3,3], [-2,1,2], [1,-1,4] );
```

The `matrix` command allows a matrix to be defined. You give each row of the matrix as a list (that is, within square brackets with the elements separated by commas).

By default, wxMaxima displays matrices using square brackets:

$$\begin{bmatrix} 2 & -3 & 3 \\ -2 & 1 & 2 \\ 1 & -1 & 4 \end{bmatrix}$$

If you are using a different interface to Maxima, or have chosen different display settings, matrices may be displayed as follows.

```
[  2  -3  3 ]
[           ]
[ -2   1  2 ]
[           ]
[  1  -1  4 ]
```

(b) If you are using wxMaxima, you can also input a matrix using an on-screen form. Use this form to assign the matrix

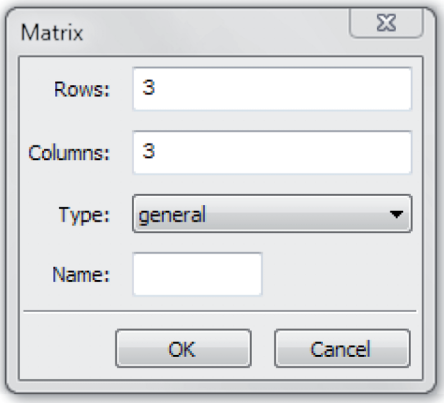
$$\begin{bmatrix} 5 & -3 \\ -1 & -2 \\ 0 & 4 \end{bmatrix}$$

to the variable `Q` as follows. (If you are not using wxMaxima, then enter this matrix using the `matrix` command, as in part (a).)

(i) From the `Algebra` menu select `Enter Matrix...`.

Make sure you select `Enter Matrix...`, not `Generate Matrix...`.

A window opens requesting properties of the matrix, as shown below.

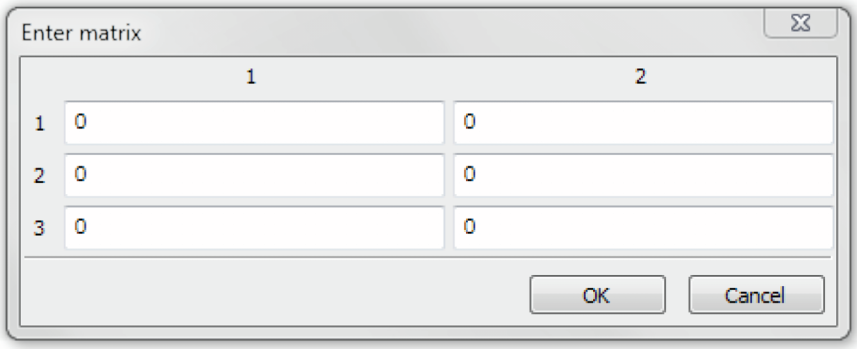


(ii) The matrix for this example has size 3×2, so enter 3 as the number of rows and 2 as the number of columns, replacing the default values shown.

It is also possible to give information on the type of matrix, but for now leave this set to `general`.

Finally, enter `Q` as the variable name to which the matrix will be assigned, and click 'OK'.

This opens a second window requesting the elements of the matrix.



(iii) Complete entry of the matrix by filling in the elements in this second window. You may like to use the `Tab` key (→|) to move the cursor between the different elements. When you have finished, click 'OK'.

Completion of this on-screen form has the effect of including a `matrix` input line in your worksheet, followed by the appropriate output.

(c) Assign the matrix $\begin{bmatrix} 1 & -2 & -1 \\ 4 & 7 & 0 \\ -5 & 1 & 3 \end{bmatrix}$ to the variable `R` using an appropriate method.

The typed command for entering a matrix is summarised below.

Entering matrices

Operation	Command	Example
Specify a matrix	`matrix(row,row,...)`	`matrix([1,2],[3,4]);`

Note: matrices can also be entered using the `Enter Matrix...` option of the `Algebra` menu.

Once a matrix has been entered, you can use it in calculations, subject to the usual laws of matrix arithmetic. The Maxima syntax to use for matrix operations is summarised in the following table. You can practise using some of these in Computer activity 19.

Note, as mentioned in Section 5, that Maxima does not strongly distinguish between column vectors and row vectors. So, for example, if $A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$ and $B = \begin{bmatrix} 8 & 9 \end{bmatrix}$ then Maxima can calculate both AB and BA, the former interpreting B as a column vector.

Matrix operations

Operation	Syntax	Example
Matrix addition	`+`	`P+Q;`
Matrix subtraction	`-`	`P-Q;`
Scalar multiple of a matrix	`*`	`2*P;`
Matrix multiplication	`.`	`P.Q;`
Matrix powers	`^^`	`P^^3;`
Transpose (of a matrix)	`transpose(matrix)`	`transpose(P);`
Determinant (of a square matrix)	`determinant(matrix)`	`determinant(P);`
Matrix inverse (of a square matrix)	`invert(matrix)` *or* `matrix^^(-1)`	`invert(P);` `P^^(-1);`

Warnings

If a matrix operation cannot be performed, for example if you attempt to add two matrices of different sizes, an error message is given.

Remember to use `.` (not `*`) for matrix multiplication and `^^` (not `^`) to find the power of a matrix.

If you use the symbol `*` when multiplying two matrices, or the symbol `^` to find a power of a matrix, then Maxima may return an answer, but in general it will be the *wrong* answer. (The symbol `*` used between two equal-sized matrices produces a matrix of the same size whose elements are the products of corresponding elements of the given matrices. The symbol `^` produces a matrix whose elements are the elements of the original matrix raised to the given power. These are sometimes called *element-wise* operations.)

Computer activity 19 Using matrix operations

In this activity,

$$\mathbf{P} = \begin{bmatrix} 2 & -3 & 3 \\ -2 & 1 & 2 \\ 1 & -1 & 4 \end{bmatrix}, \quad \mathbf{Q} = \begin{bmatrix} 5 & -3 \\ -1 & -2 \\ 0 & 4 \end{bmatrix}, \quad \text{and } \mathbf{R} = \begin{bmatrix} 1 & -2 & -1 \\ 4 & 7 & 0 \\ -5 & 1 & 3 \end{bmatrix},$$

which are the matrices you input in Computer activity 18.

(a) Find $\mathbf{P} + \mathbf{R}$.

(b) Find $3\mathbf{P}$.

> This is a scalar multiple of a matrix, so use the `*` operator.

(c) Find $\mathbf{RQ}$.

> This is matrix multiplication, so use the `.` operator.

(d) Find $\mathbf{P}^2$.

> Since $\mathbf{P}$ is a matrix use the matrix power operator `^^`.

(e) Find the determinant of $\mathbf{P}$.

(f) Find the inverse of $\mathbf{R}$.

(g) By multiplying $\mathbf{P}$ by $\begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$, show that $\begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$ is an eigenvector of $\mathbf{P}$ and find the corresponding eigenvalue.

7 Plotting graphs

You can plot various types of graph in wxMaxima using `wxplot2d` and other related commands. The '`2d`' in the name of the `wxplot2d` command arises because it produces two-dimensional plots. The '`wx`' is because it is a wxMaxima command. Commands beginning with '`wx`' insert the graphs directly into your worksheet.

If you are using another interface to Maxima, you should use commands without the '`wx`' prefix, such as `plot2d`, instead. You use `plot2d` in the same way as `wxplot2d`, and the commands have the same arguments. `plot2d` will, however, plot the graph in a different computer window, which you then need to close before you can create another plot. There is more information on this in Subsection 10.6.

The following subsections describe how to produce and customise various types of graph in Maxima. If you need to alter the default styles used for graphs, because of a disability for example, you should read Subsection 10.7 in conjunction with this section.

7.1 Plotting graphs of functions

Maxima plots graphs of functions by calculating a large number of points on the graph and joining them up.

Computer activity 20 Plotting graphs

(a) Plot the graph of the function $x \sin x$ for values of x such that $-5 \leq x \leq 5$, using the command

```
wxplot2d(x*sin(x), [x,-5,5]);
```

The first argument of the `wxplot2d` command is the expression to be plotted, and the second argument specifies the variable in the expression together with the least and greatest values that you want it to take in the graph. These are grouped in a list.

Note that wxMaxima displays the graph next to a line label beginning `%t`, such as `(%t1)`. This is how wxMaxima shows *intermediate* output, that is, output other than the final result of the command. The `wxplot2d` command has no final result, just intermediate output, which is the graph.

(b) Define the function f with the rule

$$f(t) = \frac{t^2 + 2t + 3}{t^2 + 2t - 3}$$

and plot the graph of f for values of t such that $0 \leq t \leq 3$, using the command

```
wxplot2d(f(t), [t,0,3]);
```

Notice that the first argument of the `wxplot2d` command must be $f(t)$ rather than just f and that since the independent variable is t, a range of values of t is given as the second argument.

The graph produced shows a large spike near $t = 1$. This is because $f(t)$ is not defined at $t = 1$, since its denominator is zero there.

To plot the graph, Maxima has evaluated f at values of t in the range $0 \leq t \leq 3$, obtained large values near $t = 1$ and joined up the corresponding points.

To obtain a better graph, you should restrict the range of the vertical axis. You are asked to do this next.

(c) Plot the graph of f over the range $0 \leq t \leq 3$, with the vertical range of the graph restricted to $-10 \leq y \leq 10$ using the following command.

```
wxplot2d(f(t), [t,0,3], [y,-10,10]);
```

The range of the vertical axis is restricted by including an additional argument which is a list starting with the *keyword* `y` and followed by the minimum and maximum values to be shown on the axis.

A warning:

```
plot2d:  some values were clipped.
```

is given. This is because some of the points on the graph with x-coordinates between 0 and 3 could not be displayed due to the range of values of y specified.

Warning

When plotting graphs, do not assign the expression to be plotted (or any other expression) to the *variable* `y` as this will cause an error when the keyword `y` is used in the specification of the range of the y-axis. Similarly, the variable in the expression to be plotted should not have been previously assigned to a specific value. (Remember, a variable with an assigned value can be deleted using the `kill` command.)

It is, however, possible to assign the expression to be plotted to the *function* `y`.

You can also plot multiple graphs in the same diagram by including a list of expressions as the first argument of the command, as shown in the following activity.

Computer activity 21 Plotting multiple graphs

(a) Define the functions p and q to be

$$p(x) = x\cos x \qquad \text{and} \qquad q(x) = x - \frac{x^3}{2}.$$

(b) Plot graphs of the two functions over the range $-3 \leq x \leq 3$ on the same diagram by entering:

```
wxplot2d([p(x),q(x)],[x,-3,3]);
```

Notice that the first argument of the command `wxplot2d` is a list containing the two functions to be plotted.

The curves are automatically plotted in different colours, and a *legend* showing which function corresponds to which curve is included at the top right-hand corner of the graph.

The different ways in which you can use the `wxplot2d` command are summarised below.

Plotting graphs

Operation	Command	Example
Plot a graph	`wxplot2d(expression, horizontal range,...)`	`wxplot2d(x^2,[x,0,1]);` which plots the graph of x^2 for x between 0 and 1 `wxplot2d(x^2,[x,0,1],[y,0,2]);` which plots the graph of x^2 for x between 0 and 1, and with vertical axis values between 0 and 2
Plot several graphs	`wxplot2d(list of expressions, horizontal range,...)`	`wxplot2d([x^2, 2*x],[x,0,1]);` which plots graphs of x^2 and $2x$ for x between 0 and 1

7.2 Changing the appearance of a graph

In this subsection some of the commands needed to change various aspects of the appearance of a graph are reviewed.

Changing graph proprieties

Maxima lets you control many properties of the graph, for example the colour of the curves plotted and the text in the legend. This is done by including additional arguments in the `wxplot2d` command, each being a list beginning with a keyword describing the property to be changed and followed by the required value of the property. Where properties apply to individual curves, a value should be listed for each curve plotted. This is demonstrated in the following activity.

Computer activity 22 Changing the properties of a graph

Plot the graph of the functions p and q from Computer activity 21 again, this time with

- p coloured black and q green
- the graph legend changed to refer to the names p and q rather than the expressions they represent
- the vertical axis labelled y

by entering

```
wxplot2d([p(x),q(x)], [x,-3,3], [color, black, green],
[legend, "p", "q"], [ylabel, "y"]);
```

The additional settings are given as lists, each beginning with a keyword.

The colours of the curves are given in a list beginning `color` (American spelling), and one colour is listed for each curve. Maxima uses the first colour listed for the graph of the first expression listed and so on. If you do not include enough colours, Maxima starts using the listed colours again.

The list beginning `legend` specifies the legend text for each curve, each enclosed in double quote marks.

The list beginning `ylabel` sets the text to be displayed on the vertical axis. If required, text for the horizontal axis label can be specified using a list beginning with the keyword `xlabel`.

The legend can be moved to the bottom right-hand corner of the graph by including the additional argument

```
[gnuplot_preamble, "set key bottom"]
```

(Gnuplot is the underlying software that Maxima uses to plot graphs.)

The legend can be suppressed by including the argument

```
[legend, false]
```

Changing axis scales

Whatever ranges of x- and y-values you specify for a plot, by default Maxima makes the y-axis appear $\frac{3}{5}$ as long as the x-axis.

In some cases, however, it is better to plot the graph with equal scales on each axis. You can do this by including the additional argument

```
same_xy
```

in the `wxplot2d` plotting command.

Note that in some older releases of Maxima, the option needed to specify equal scalings on coordinate axes was

```
[gnuplot_preamble, "set size ratio -1"]
```

If the `same_xy` option does not change the scales of your graph, then use this option instead.

Computer activity 23 Plotting a graph with equal scales

Plot the graph of $y = 2x^2$ using equal scales on the x- and y-axes by entering

```
wxplot2d(2*x^2,[x,-1,1], same_xy);
```

If this command does not have the desired effect, try using

```
wxplot2d(2*x^2,[x,-1,1],
[gnuplot_preamble, "set size ratio -1"]);
```

instead.

Including the `same_xy` (or `gnuplot_preamble`) argument in the `wxplot2d` command can be tiresome if you are plotting lots of curves and need equally scaled x- and y-axes for each. An alternative is to change the default behaviour of Maxima so that it uses axes with equal scales for the rest of your session. You will see how to do this in the following activity.

Computer activity 24 Plotting another graph with equal scales

Plot the graph of $\sin(x)$ for $-\pi \leq x \leq \pi$, using equal scales on the x- and y-axes as follows.

(a) Tell Maxima to always use equal scales on the x- and y-axes by entering

```
set_plot_option(same_xy)$
```

This sets `same_xy` to be a default option for all plots during the rest of your Maxima session. The line ends with `$` to prevent the display of the output of the command. Without this, the output consists of a long list of all the current default settings, which can safely be ignored.

If you are using an older release of Maxima that uses the `gnuplot_preamble` option, use

```
set_plot_option([gnuplot_preamble, "set size ratio -1"])$
```

instead.

(b) Plot the graph by entering

```
wxplot2d(sin(x),[x,-%pi,%pi]);
```

If you wish to stop plotting graphs with equal scales and restore the default Maxima plotting behaviour, then set the `same_xy` plotting option to be `false` using

```
set_plot_option([same_xy, false])$
```

If you are using an older version of Maxima, the command needed to reset the default view is

```
set_plot_option([gnuplot_preamble, ""])$
```

Changing the graph size

Finally, the size of the plot within your worksheet can be changed using the `wxplot_size` system variable. For example, entering

```
wxplot_size:[800,600];
```

changes the size of subsequently plotted graphs from the default setting of 600×400 pixels to 800×600 pixels.

Summary

Graph plotting options

Option	Argument	Example
Vertical range	`[y, , ]`	`[y, 0, 5]`
Curve colour	`[color, , ...]`	`[color, red]`
Legend text	`[legend, " ", ...]`	`[legend, "Car A"]`
Turn legend off	`[legend, false]`	`[legend, false]`
Position legend at the bottom of the graph	`[gnuplot_preamble, "set key bottom"]`	`[gnuplot_preamble, "set key bottom"]`
Position legend to the right of the graph	`[gnuplot_preamble, "set key outside"]`	`[gnuplot_preamble, "set key outside"]`
Horizontal axis label	`[xlabel, " "]`	`[xlabel, "t (h)"]`
Vertical axis label	`[ylabel, " "]`	`[ylabel, "s (km)"]`
Set axes to have equal scales	`same_xy`	`same_xy`
Set axes to have equal scales *(in older versions of Maxima)*	`[gnuplot_preamble, "set size ratio -1"]`	`[gnuplot_preamble, "set size ratio -1"]`
Change the size of the graph	`wxplot_size:[ , ]`	`wxplot_size:[800,600];`

Note: for a full list of the possible options, view the Maxima help obtained using the `?? plotting options;` command.

Setting default plotting options

Operation	Command	Example
Set the default plotting options	`set_plot_option(option)`	`set_plot_option(same_xy)$`

7.3 Plotting graphs of discrete data

Maxima can also plot graphs of discrete data. To do this, it is convenient to store the coordinates of the points to be plotted in a list, as shown in the following activity.

Computer activity 25 Plotting discrete data

Plot the graph of the points given by

$$(i, i^2) \quad (i = 1, 2, \ldots, 10)$$

as follows.

(a) First, create a list of the points to be plotted, and assign it to the variable `pts` by entering

```
pts:makelist([i,i^2], i, 1, 10);
```

This uses the `makelist` command reviewed in Section 4. It creates a list whose elements are pairs (themselves stored in a list) of the x- and y-coordinates of points to be plotted.

Do not use the variable name `points` for your list of points: this is a word used by Maxima to set plot styles, as you will see later.

(b) Plot the graph of the points using the command

```
wxplot2d([discrete, pts]);
```

Here, instead of the first argument of the `wxplot2d` command being an expression, or a list of expressions, it is a list containing two elements. The first element is the keyword `discrete`. This tells Maxima that a set of discrete points are to be plotted. The second element is the list of the coordinates of the points to be plotted.

By default, the points plotted are joined by a line. When plotting discrete data it is more usual to only plot the points, and not join them by a line. You can tell Maxima to do this by including an additional argument to the `wxplot2d` command, as you will do next.

(c) Plot the graph again, only showing the points by entering

```
wxplot2d([discrete, pts], [style, points]);
```

Other useful graph styles are listed in the Graph plotting options table at the end of this subsection.

You can also include other arguments of `wxplot2d` that you met earlier, such as changing the colour of the points, and the axis limits.

Note that graphs of (several) sets of discrete points and graphs of (several) functions can all be plotted on the same figure, by including the equations of the functions, and the specifications of the points (of the form `[discrete, pts]`) in a list as the first argument of the `wxplot2d` command.

The use of `wxplot2d` for plotting discrete points is summarised below.

Plotting discrete data

Operation	Command	Example
Plot a set of points	`wxplot2d([discrete, list of coordinate pairs], [style, points], ...)`	`wxplot2d([discrete, [[1,5], [2,10]]], [style, points]);`

Graph plotting options

Option	Argument	Example
Plot points	`[style, points]`	`[style, points]`
Plot a line and points	`[style, linespoints]`	`[style, linespoints]`

8 Solving equations

Maxima can solve equations either analytically (when possible) or by finding numerical approximations to solutions. Equations are written in Maxima using an equals sign in the usual way.

8.1 Solving equations analytically

You can find the exact solution(s) of equations, and change the subject of an equation, using the Maxima `solve` command. This solves equations using algebraic manipulations. The command is demonstrated in the following activity.

Computer activity 26 Solving and rearranging equations

(a) Solve the equation $x^3 - 4x^2 + 2x + 4 = 0$ by entering

```
sol:solve(x^3-4*x^2+2*x+4=0);
```

Notice that Maxima returns the solutions of the equation as a list, which has here been assigned to the variable `sol`. Each element of the list is an equation with the variable on the left-hand side and its value that satisfies the equation on the right.

The value of the first solution of this equation can be obtained by entering

```
rhs(sol[1]);
```

which gives the right-hand side of the first element of the list `sol`.

(b) Rearrange the equation $y = \ln(ux + v)$ to make x the subject by entering

```
solve(y=log(u*x+v), x);
```

Remember, the Maxima function for ln is `log`.

The second argument of the `solve` command gives the variable we wish to 'solve' for.

(c) Solve the equation $\tan(3x - 1) = \sqrt{3}$.

When solving equations involving trigonometric functions Maxima will only try to find one solution, even though many others may exist.

The warning

```
solve:  using arc-trig functions to get a solution.
Some solutions will be lost.
```

is given to alert you to this.

(d) Solve the simultaneous equations

$$y = x^2 - 6x + 9$$
$$(x - 3)^2 + (y - 2)^2 = 3$$

by entering

```
solve([y=x^2-6*x+9, (x-3)^2+(y-2)^2=3], [x,y]);
```

To solve simultaneous equations, enter the equations as a list as the first argument of `solve`, and list the unknowns to solve for as the second argument.

Here there are four solutions. Maxima gives a list of each pair of values of x and y which satisfy the equations.

(e) Try to use Maxima to solve the equation $x^9 + 2x - 1 = 0$.

Here, Maxima returns (within a list) the equation, slightly rearranged, but unsolved.

The `solve` command always tries to find an exact solution of the equation using algebraic manipulations. If it is unable to do this, then it returns the original equation, as here.

In the next subsection you will revise how to find approximate numerical solutions of such equations.

The use of the `solve` command illustrated in the previous activity is summarised below.

Solving equations and changing the subject of an equation

Operation	Command	Example
Solve an equation exactly	`solve(equation)`	`solve(2*x^2-1=0);`
Solve an equation for a variable, i.e. change the subject	`solve(equation, variable)`	`solve(2*a*b-3*b=0, a);`
Solve simultaneous equations	`solve(list of equations, list of variables)`	`solve([x^2+y=1, y^2-x=1], [x,y]);`

Extracting the left- and right-hand sides of equations

Operation	Command	Example
Left-hand side of an equation	`lhs(equation)`	`lhs(4*x+1=2*x-2);`
Right-hand side of an equation	`rhs(equation)`	`rhs(4*x+1=2*x-2);`

8.2 Solving equations numerically

In Computer activity 26(e) you saw an example where the `solve` command failed to find the exact solution of an equation. In such circumstances you can use other Maxima commands, such as `find_root` or `allroots`, to try to find an approximate numerical solution(s). The `find_root` command seeks a single real solution of an equation. The `allroots` command seeks all solutions (real and complex) of a *polynomial* equation; it does not accept non-polynomial equations.

Computer activity 27 Solving equations using `find_root`

Find a numerical approximation to a solution of the equation $x^9 + 2x - 1 = 0$ seen in Computer activity 26(e) as follows.

(a) First, define the function f in Maxima, with the rule

$$f(x) = x^9 + 2x - 1.$$

(b) Next, plot the graph of $f(x)$ over the interval $-1 \leq x \leq 1$, say. This will help you to find an interval that contains a solution.

From the graph, you can see that $f(0) < 0$ and $f(1) > 0$, so there is a solution in the interval $(0, 1)$.

If there were no solution in the interval over which you plotted the function, then you could plot the function over a larger interval, and so on, until you find a suitable interval, if one exists.

(c) Find a numerical approximation to the solution of the equation in the interval $(0, 1)$ by entering

```
find_root(f(x), x, 0, 1);
```

Remember to type the underscore symbol (_) in the middle of the command name.

The `find_root` command looks for a single real solution of an equation of the form $f(x) = 0$ that lies within an interval that you specify. The command requires that the values of f at each endpoint of the interval have different signs. If this is not the case, then Maxima gives an error message.

Notice that the first argument of the `find_root` command is an *expression* (or a function defined by an expression) not an *equation*.

Computer activity 28 Solving equations using `allroots`

Find numerical approximations to all the solutions of the equation $x^9 + 2x - 1 = 0$ by entering

```
allroots(x^9+2*x-1=0);
```

Maxima returns a list of all 9 real and complex roots of this degree 9 polynomial. The only real root is that found in the previous activity.

The `find_root` and `allroots` commands are summarised below.

Solving equations numerically

Operation	Command	Example
Find (numerically) a root of an equation within a given interval	`find_root(expression,variable, interval start value, interval end value)`	`find_root(2*x^3-1,x,0,1);`
Find (numerically) all roots of a polynomial equation	`allroots(equation)`	`allroots(2*x^3-1=0);`

9 Calculus

The Maxima commands for differentiating and integrating expressions are `diff` and `integrate`. These are demonstrated in the following activity.

Computer activity 29 Differentiating and integrating

(a) Find the derivative of $f(x) = \dfrac{\cos x}{x^2+3}$, by entering

```
diff(cos(x)/(x^2+3), x);
```

> The first argument of the `diff` command is the expression to be differentiated, and the second argument is the variable to differentiate with respect to.
>
> You must include the second argument. If you do not, then the answer returned by Maxima will be the derivative multiplied by `del(x)`, which represents the differential of `x`.
>
> As you would expect, the first argument can also be a variable that you have previously defined to be an expression, or a function you have previously defined.

(b) Find the second derivative of $g(u) = u^2 \sin u$, by entering

```
diff(u^2*sin(u), u, 2);
```

> The third argument of `diff` specifies the number of times that the expression is to be differentiated. (If this is omitted, as in part (a), then the first derivative is calculated.)

(c) Find the third derivative of $h(t) = \dfrac{\ln t}{t^2}$.

> Remember, the Maxima function for ln is `log`.

(d) Evaluate $\int x \sin^2(x)\, dx$ by entering

```
integrate(x*sin(x)^2, x);
```

> The first argument of `integrate` is the expression to integrate, and the second is the variable to integrate with respect to.

The output returned by Maxima does not include an arbitrary constant. That is, Maxima gives an antiderivative of the expression, not its indefinite integral. When you write down an indefinite integral that you have found using Maxima, you need to remember to add an arbitrary constant yourself.

(e) Evaluate the definite integral $\int_0^1 t^2 e^t \, dt$ by entering

```
integrate(t^2*%e^t, t, 0, 1);
```

The third and fourth arguments of `integrate` are the lower and upper limits of integration, respectively.

Notice that, as usual, Maxima returns an exact result. Use the `float` command to find a decimal approximation, if required.

(f) Evaluate the definite integral $\int_1^2 \frac{1}{x^2} \, dx$

(g) Try to evaluate $\int_0^1 \tan(e^{-x}) \, dx$

Maxima cannot calculate this integral, so it is displayed unevaluated.

As you saw in Computer activity 29(g) it is not always possible to express the result of an integral exactly. Sometimes, Maxima may give a result in terms of the so-called *special functions*. These cannot be simply expressed in terms of simple functions, but many can be defined in terms of antiderivatives of them. Such functions include `erf`, which is known as the *Error function*, and `gamma_incomplete`, the *incomplete gamma function*.

The `diff` and `integrate` commands are summarised below.

Calculus

Operation	Command	Example
Differentiate	`diff( , variable )`	`diff(sin(x^2),x);`
Differentiate multiple times	`diff( , variable , positive integer )`	`diff(log(x),x,3);` to differentiate three times
Integrate (find an antiderivative)	`integrate( , variable )`	`integrate(x^2,x);`
Evaluate a definite integral	`integrate( , variable , lower limit , upper limit )`	`integrate(sin(x),x,0,1);`

If you wish to assign the result of a `diff` or `integrate` command to a named function, then you need to take care that the function is defined to be equal to the *result* of the operation, not the operation of differentiating or integrating itself.

This is demonstrated in the following activity.

Computer activity 30 Defining a function to be the derivative or integral of another function

(a) Define the function $f(x) = x\sqrt{1+4x^2}$.

(b) Try to define `df(x)` to be the derivative of $f(x)$, by entering

```
df(x):=diff(f(x),x);
```

(c) Try to find the value of the derivative of f at $x = 3$ by evaluating `df(x)` at $x = 3$; that is, by entering

```
df(3);
```

This gives an error. The problem is that when you entered the command

```
df(x):=diff(f(x),x);
```

Maxima did not perform the differentiation. Instead, it just noted that `df(x)` means `diff(f(x),x)`. Then, when you entered `df(3)`, Maxima understood this to mean `diff(f(3),3)`, which is meaningless, as you cannot differentiate with respect to a particular number.

To correctly define `df(x)` to be the *result* of differentiating `f(x)`, you need to force Maxima to perform the differentiation at the point `df(x)` is defined.

You can do this using the syntax `''(  )` (which contains two single quotation marks, not a double quotation mark, and a pair of round brackets). This is demonstrated below.

(d) Define `df(x)` to be the *result* of differentiating f with respect to x, by entering

```
df(x):=''(diff(f(x),x));
```

Remember that `''` is two *separate* quotation marks. Use the upright quotation mark on your keyboard (`'`), not the sloping one (`` ` ``). Don't forget to include the outer round brackets too.

Notice that this time the output line shows that `df(x)` is indeed defined to be the result of differentiating `f(x)`.

(e) Find the value of the derivative of f at $x = 3$ by entering

```
df(3);
```

(f) Use a similar approach to define `F(x)` to be an antiderivative of `f(x)`.

(g) Calculate `F(1)-F(0)` and confirm this is the same value as $\int_0^1 f(x)\,dx$.

The command used to force the evaluation of another command is summarised overleaf.

Forcing the evaluation of a command

Operation	Command	Example
Force the evaluation of a command	''( )	g(x):=''(diff(x^4,x));

Assumptions

Sometimes you may want to integrate a function that contains a variable other than the one you are integrating with respect to. For example, the integral $\int x^n\,dx$ contains the variable n as well as the variable x. When you use Maxima to find such an integral, you sometimes need to provide information about the other variable as shown in the next activity.

Computer activity 31 Giving Maxima more information

(a) Try to use Maxima to find the integral $\int x^n\,dx$, as follows.

(i) First, ensure that `n` has no predefined value, by entering

```
kill(n);
```

(ii) Then, enter

```
integrate(x^n, x);
```

Maxima asks the question: `Is n equal to -1?`

This is because the answer to $\int x^n\,dx$ depends on the value of n.

If $n = -1$, then the integral is $\int \frac{1}{x}\,dx$, which is equal to $\ln|x| + c$, whereas if n is any other value, then the integral is equal to $\frac{x^{n+1}}{n+1} + c$. So Maxima cannot calculate the integral without knowing more about n.

Enter the answer `n;`
(which is short for `no`)

You could also answer the question with `y;` to mean `yes`.

Maxima displays the expected result.

(b) An alternative approach is to give Maxima information about n *before* you use the `integrate` command. You can do this as follows.

(i) Tell Maxima that $n \neq -1$ by entering

```
assume(notequal(n,-1));
```

The `assume` command tells Maxima information about variables. For example, `assume(a>0);` tells Maxima that `a` is positive.

Maxima has no simple symbol for 'not equal to', so you have to use the `notequal` command for this, as above.

(ii) Now integrate x^n using

```
integrate(x^n, x);
```

No question is asked this time, since Maxima knows all it needs to know about `n`.

(iii) Enter

```
facts();
```

to see all the additional facts about variables known to Maxima.

This lists all such facts. The only one shown should be the one you gave earlier.

(iv) Tell Maxima to forget about the assumption on `n` by entering

```
forget(notequal(n,-1));
```

(v) Type `facts();` again to see all the facts now known.

No facts are displayed, since Maxima has now forgotten the assumption about n.

The commands used in the previous activity are summarised below, together with the `equal` command, which is used in a similar fashion to `notequal`.

Facts about variables

Operation	Command	Example
Make an assumption about a variable	`assume( )`	`assume(a>0);`
State two things are equal	`equal( , )`	`assume(equal(n,3));`
State two things are not equal	`notequal( , )`	`assume(notequal(n,-1));`
Forget a property	`forget(property)`	`forget(a>0);`
List all known facts	`facts()`	`facts();`
List all known facts about a particular variable	`facts(variable)`	`facts(a);`

10 Accessibility

This section outlines issues associated with the use of Maxima if you interact with a computer using only a keyboard, use a screenreader or require different colour settings. This information should help you decide whether the wxMaxima interface is suitable for you, or whether you should consider using an alternative interface, such as the command-line interface. It also explains how to configure Maxima to make it easier for you to use and contains alternative versions of Subsection 1.2 and parts of Section 2 describing how to use the command-line interface.

Activities within this section are numbered to match corresponding activities in Sections 1 and 2. If you decide to use the command-line interface, the format of the solutions given at the end of this guide may not exactly match the output you obtain.

10.1 Choosing and configuring your interface

This subsection describes how wxMaxima can be configured to help those with additional requirements. If you find these changes are not sufficient, you may wish to use the command-line interface to Maxima instead.

Using wxMaxima with a keyboard alone

The majority of the wxMaxima interface is accessible using a keyboard alone. On a Windows computer you can access the wxMaxima menus by pressing and releasing the Alt key, navigating through the menus using the keyboard arrow keys, and then pressing Enter to make your selection. Alternatively, you can press and hold down the Alt key, and then press the keyboard key that corresponds to the underlined letter in a menu entry to select it. In both cases, you can leave menu navigation without making a selection by pressing the Esc key.

On an Apple Mac you can access the wxMaxima menus by pressing and releasing the F2 key while holding down the Ctrl and Fn keys (or just Ctrl on some keyboards), navigating through the menus using the keyboard arrow keys, and then pressing Enter to make your selection. You can leave menu navigation without making a selection by pressing the Esc key.

The wxMaxima toolbar cannot be accessed using the keyboard alone, but this is not used extensively in this module and alternative typed commands for accessing the functions are given as required.

Changing wxMaxima colours and fonts

You can change the appearance of the text used by wxMaxima as follows. First, click on the following toolbar icon.

Alternatively, select `Configure` from the `Edit` menu (or, on an Apple Mac, select `Preferences` from the `wxMaxima` menu).

In the configuration window that opens, select the following 'Style' icon.

You can then change the colour, style, and sometimes the font and size, of individual elements of the wxMaxima interface by first choosing the appropriate element from the 'Styles' list, and then selecting your preferred settings.

Where a selected element has a greyed-out 'Choose font' button this is usually because it shares its font with other elements of the interface. For example, those elements that form part of the mathematical output of the interface share a font that you can change by first clicking on the Math font 'Choose font' button and then selecting your preferred font and size. Most of the remaining elements with a greyed-out font button share a font that you can change in a similar way by clicking on the Default font 'Choose font' button. If you choose a large font size, some horizontal scrolling may be necessary to read all the text displayed in a worksheet.

Unfortunately, the Configuration window is not keyboard accessible, and so if you only use a keyboard you may need the assistance of a non-medical helper to initially configure the system.

The Configuration window does not change the colours and fonts used in graphs produced by Maxima. Subsection 10.7 explains how to change the colours, fonts and line thickness used in graphs. You should read Subsection 10.7 in conjunction with Section 7, which discusses how to plot graphs.

Using a screen magnifier or screenreader

The text in wxMaxima can become pixelated when you use a screen magnifier. An alternative is to increase the size of the text font as described above and/or use the zooming facilities on the `View` menu, which you can also access using `Alt-I` (zoom in) and `Alt-O` (zoom out).

The wxMaxima interface is not accessible to a screenreader. If you use a screenreader you might wish to consider using the command-line interface to Maxima instead of the wxMaxima interface. You should ensure your screenreader is set to read all the punctuation. Details of using Maxima with the command-line interface are given below. The graphs produced by Maxima are not accessible to a screenreader. You might need assistance when using these.

10.2 Starting, testing and configuring Maxima using the command-line interface

This subsection describes how to start, test and configure Maxima when using the command-line interface. If you are using the command-line interface you should read this subsection instead of Subsection 1.2, which this replaces.

Computer activity 2 Starting Maxima

Start the command-line interface to Maxima on your computer and keep it open to work with as you study this section.

If you cannot remember how to do this on your computer, see the OU Maxima website: `learn1.open.ac.uk/site/maxima`.

The command-line interface to Maxima is illustrated in Figure 6.

At the top of the window is some introductory text, followed by

```
(%i1)
```

This is how Maxima labels lines. The `i` stands for 'input': this is input line number 1 and is where you can type your first command. These commands instruct Maxima to either perform calculations or change system settings.

The percentage symbol (`%`) is used by Maxima to indicate objects built into the system. There is more about this later.

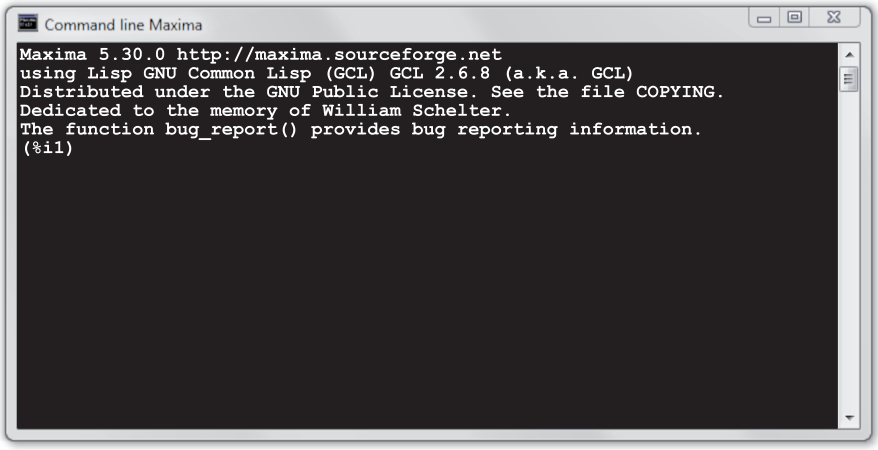

Command line Maxima
Maxima 5.30.0 http://maxima.sourceforge.net
using Lisp GNU Common Lisp (GCL) GCL 2.6.8 (a.k.a. GCL)
Distributed under the GNU Public License. See the file COPYING.
Dedicated to the memory of William Schelter.
The function bug_report() provides bug reporting information.
(%i1)

Figure 6 The initial command-line interface window

The following activity shows you how to use Maxima to perform a simple calculation and checks that your system is working properly. Follow the steps using Maxima as you read the activity.

If you encounter unexpected problems when working through this activity, check the Frequently asked questions (FAQs) section on the OU Maxima website: `learn1.open.ac.uk/site/maxima`.

Computer activity 3 Your first Maxima calculation

Follow these steps to use Maxima to calculate $2 + 3 \times \frac{4}{5}$.

(a) Click anywhere on the Maxima window, to ensure that the text you type next is directed to the correct place.

(b) Type the following exactly as it appears here:

```
2+3*4/5;
```

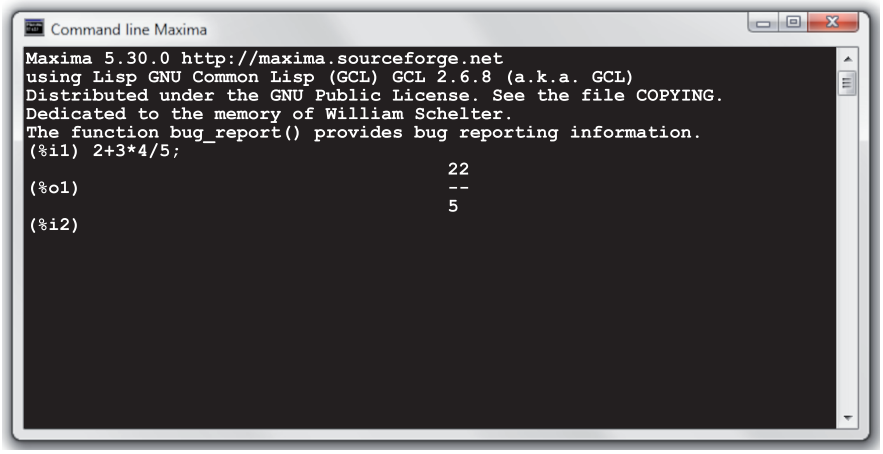
including the semicolon (`;`) at the end.

> The text you type will appear next to the input line number `(%i1)`, as indicated by a flashing horizontal line known as the **editing cursor**.

(c) Press Enter.

> If you forget to type the semicolon (`;`) at the end of the command, the editing cursor simply moves to the next line, waiting for you to type more. Maxima will continue to do this until you indicate you have reached the end of the command by typing a semicolon. So if you forgot the semicolon, then type it now and press Enter again.
>
> Pressing Enter instructs Maxima to calculate the expression. This is known as **evaluating** the expression.
>
> The result is displayed under the input line, preceded by the label `(%o1)`. The `o` stands for 'output', so this label identifies the line as output line 1. It gives the output corresponding to input line 1. Throughout each Maxima session the line numbering starts at 1 and then increases by one for each new input (and output).

Maxima then displays another input line number, ready for you to enter your next command.


Command line Maxima
Maxima 5.30.0 http://maxima.sourceforge.net
using Lisp GNU Common Lisp (GCL) GCL 2.6.8 (a.k.a. GCL)
Distributed under the GNU Public License. See the file COPYING.
Dedicated to the memory of William Schelter.
The function bug_report() provides bug reporting information.
(%i1) 2+3*4/5;
22
(%o1) --
5
(%i2)

If, in Computer activity 3, Maxima behaved as described, then you have a working system to use throughout MST210.

The result of the calculation in Computer activity 3 was displayed as formatted mathematics over several lines, replicating how mathematics is usually written. This can, however, cause problems if you are using a screenreader. The display behaviour can be changed for every Maxima session as described in the following activity.

Computer activity 4 Configuring the Maxima display of mathematics

You should only do this activity if you wish to change how Maxima displays mathematics.

(a) First exit Maxima by entering the command

```
quit();
```

(on some installations you may also have to close the window).

(b) Turn off the two-dimensional display of mathematics by adding the Maxima command `display2d:false;` to your `maxima-init.mac` configuration file, as follows.

(i) Follow the instructions in the Accessibility section of the OU Maxima website `learn1.open.ac.uk/site/maxima` to locate your `maxima-init.mac` configuration file, or create one in the correct location as appropriate.

(ii) Open your `maxima-init.mac` file in a plain text editor (such as Notepad on a Windows computer, or TextEdit on an Apple Mac), and add the text

```
display2d:false;
```

on a new line. There are further details on how to do this in the Accessibility section of the OU Maxima website.

(iii) Save your modified `maxima-init.mac`, and close your text editor.

(c) Restart Maxima and enter the calculation from Computer activity 3 again, that is, enter

```
2+3*4/5;
```

Notice that Maxima now displays the result on a single line:

```
22/5
```

Another useful configuration change is to make Maxima display the result of a command left-justified against the output line number, instead of the default behaviour of displaying it centred on the line. This can be achieved by entering the command

```
leftjust:true;
```

either in the Maxima window, or, if you wish to make the change permanent, in your `maxima-init.mac` file as described above.

Closing wxMaxima

When you have finished using Maxima, close it by using one of the following methods.

- Enter `quit();`
- On a Windows computer, click the usual small cross button at the top right-hand corner of the Maxima window.

10.3 Troubleshooting Maxima using the command-line interface

If you are using the command-line interface to Maxima and it does not seem to be responding, try the following suggestions.

Troubleshooting Maxima

- Check you have ended your previous command with either a semicolon (`;`) or a dollar (`$`). Maxima will not do anything without them!
- Check that Maxima is not waiting for you to type something. Is a question being displayed that you have not yet answered?
- If Maxima is taking too long to evaluate a command, you can interrupt it by pressing `Ctrl-C`.

  If you do this, you may see the following output, or something similar, which you can safely ignore.

```
Maxima encountered a Lisp error:

  EXT:GC: User break

Automatically continuing.

To enable the Lisp debugger set *debugger-hook* to nil.
```

Note that pressing `Ctrl-C` at other times may close your Maxima session.

If you encounter other problems when using Maxima, check the Frequently asked questions (FAQs) section of the OU Maxima website: `learn1.open.ac.uk/site/maxima`.

10.4 Reusing and editing commands in the command-line interface

When using the command-line interface, you can edit and re-evaluate previously entered commands or calculations, as shown in the following activity.

Computer activity 6 Editing a command

(a) Calculate $2\sqrt{3}+5\sqrt{27}$ by entering the following line into Maxima.

```
2*sqrt(3)+5*sqrt(27);
```

(b) Edit your entry in part (a) to calculate $2\sqrt{3}-5\sqrt{27}$ as follows.

(i) Press the keyboard up arrow key once, so that the previously entered command appears against the current input line number (on an Apple Mac you may need to hold down the Alt key while pressing the keyboard up arrow key once).

> You can access all the commands you have previously entered in your Maxima session using the up and down keyboard arrow keys.

(ii) Edit the expression by moving the editing cursor using the left and right keyboard arrow keys, typing new text and using the ← (backspace) key to make deletions.

(iii) Press Enter.

Maxima also permits the result of one calculation to be used in another calculation. The symbol `%` represents the result of the last evaluated command. So, for example, if you enter `sqrt(%);` Maxima calculates the square root of the last evaluated output. You can also use the output (or input) of other lines, by typing the line label in a calculation. For example, if you enter `%o5^3;` Maxima calculates the cube of the expression in output line 5.

Computer activity 7 Using the results of previous calculations

(a) Enter `sin(5*%pi/6);` into Maxima and evaluate it.

(b) On the following line type `2*%;` and press Enter.

> Maxima returns `1`, which is twice the previous answer.

(c) Enter `%o` ▢ `^2;` with ▢ replaced by the output line number corresponding to the result of `sin(5*%pi/6);` entered earlier.

This command finds the square of the result of the line referenced.

10.5 Saving and printing your work using the command-line interface

After working with Maxima you will probably want to save your calculations so that you can reuse or read them at a later date.

When you are using the command-line interface you cannot save the session directly, as you do when saving other files. Instead there are two stages in saving your work:

- If you want to keep a record of the contents of the command-line window, then you need to save a *transcript* of the session. You can then print this transcript if needed.
- If you want to save the values of any variables and functions you have defined, so that you can continue working with them in a new Maxima session, then you will need to save the current state of the system. Note that this does not give you access to the commands used to define the variables or functions, so you cannot edit them in the new session.

These two processes are illustrated in Computer activity 8.

When saving your work, you need to tell Maxima exactly where to save the file containing the session state or your transcript by giving the complete *pathname* of the file, which includes not only the name of the file but also the hierarchy of folders in which the file is located. Unless you do this, Maxima will try to save the file in the location in which the system is installed on your computer. Not only is this not recommended, but as a computer user, you may not have sufficient permission to do this.

The pathname of a file depends on its location, the type of computer you are using, and your computer username. In the following activity, we assume that a user with the computer username 'bill' wishes to save a transcript of his Maxima session in a file called `transcript.txt` within a folder called `maxima` which he created in Computer activity 1. The `maxima` folder is within Bill's 'Documents' area on a recent Windows computer (that is, Windows Vista, 7, 8 or 10). In this case, the full pathname of his file is

```
C:/Users/bill/Documents/maxima/transcript.txt
```

(Notice that Maxima separates parts of the pathname with a forward slash (/) rather than the usual Windows backward slash (\).)

If Bill were using an older Windows computer (such as Windows XP), the equivalent pathname would be

```
C:/Documents and Settings/bill/Documents/maxima/transcript.txt
```

Using an Apple Mac the equivalent is

```
/Users/bill/Documents/maxima/transcript.txt
```

Computer activity 8 Saving your work

(a) Create a transcript of your current session by following these steps.

(i) Enter

```
writefile("C:/Users/bill/Documents/maxima/transcript.txt");
```

replacing the pathname with one appropriate to you and your system.

The path of the file is given in double quote marks.

The command tells Maxima to save a copy of all the subsequent input and outline lines to the file named `transcript.txt` in your chosen folder.

The output

```
#<OUTPUT BUFFERED FILE-STREAM CHARACTER
C:\Users\bill\Documents\maxima\transcript.txt>
```

done

is given, where the pathname is the one you gave in the command.

(ii) Enter

```
playback();
```

This command redisplays each of the input and output lines of your session, and when used in conjunction with the `writefile` command results in a copy of these lines being stored in your chosen file.

To playback only a selection of the lines from your session, you can use a command of the form

```
playback([first line number, last line number]);
```

which plays back only those lines within the given range.

(iii) Enter

```
closefile();
```

This stops the recording of the session into the file.

The output

```
#<CLOSED OUTPUT BUFFERED FILE-STREAM CHARACTER
C:\Users\bill\Documents\maxima\transcript.txt>
```

done

is given.

(iv) Find the file `transcript.txt` on your computer and open it using a text editor, for example, Microsoft Word.

The file contains a copy of the input and output lines from your session. (When opened in some text editors the file may appear all on one long line.)

(b) Save your current session then reload it, by following these steps.

(i) Enter
`save("C:/Users/bill/Documents/maxima/mysession.lisp", all);`
replacing the pathname with one appropriate to you and your system.

This tells Maxima to save your current session to the file `mysession.lisp` in your chosen folder. (Lisp is the computer programming language used to create Maxima, and the `save` command saves your work using commands from this language.)

The path of the file is output.

(ii) Close Maxima by entering

```
quit();
```

(iii) Restart Maxima in the usual way.

(iv) Load your previously saved session by entering
`load("C:/Users/bill/Documents/maxima/mysession.lisp");`
using the same pathname you used in part (b)(i).

This tells Maxima to restore the state of the system to that saved in the specified file.

(v) Enter

```
values;
```

to display the names of all known variables.

You should see that all the variables from your previous session, if any, are known. You might like to check their values are as expected too.

There is more about variables and the `values` command in Subsection 3.1.

An alternative way of using the `writefile` and `closefile` commands is to enter the first at the very start of your session and the second at the end, so that everything you do is recorded.

Saving your work

Operation	Command
Start writing a transcript	`writefile("file path");`
Stop writing a transcript	`closefile();`
Playback all the input and output in a session	`playback();`
Playback a range of input and output	`playback([start line number, end line number]);`
Save the state of a session	`save("file path", all);`
Reload a session	`load("file path");`

You can also obtain an image of the Maxima window by taking a 'screenshot'. To do this on a Microsoft Windows computer, first click on the Maxima window, then press `Alt-PrintScreen` (that is, hold down the Alt keyboard key while pressing the `PrintScreen` key, which may be labelled PrtScr , Prt Scrn or something similar). This stores the image of the Maxima window, which you can then paste into a suitable application. If you are using Windows Vista, Windows 7, 8 or 10, you might like to use the 'Snipping Tool', available by typing 'snip' into the Start menu.

To take a screenshot on an Apple Mac computer, press `Cmd-Ctrl-Shift-4` (that is, hold down the Command , Ctrl and ⇧ keyboard keys while pressing the 4 key). Next press the space bar, move the camera pointer over the Maxima window and click. This stores the image of the Maxima window, which you can then paste into a suitable application. You may also like to use the Grab application from the Utilities folder.

10.6 Plotting graphs without wxMaxima

When using the command-line interface, you should use the `plot2d` command rather than the `wxplot2d` command described in Section 7. These commands work similarly, and have the same arguments. The main difference is that `plot2d` opens a new computer window containing the graph.

The graph window can be closed by pressing the usual cross at the top right-hand corner, or by pressing `Alt-F4`. You will not be able to enter any commands into the Maxima command-line interface until the graphing window is closed.

The graph can be saved as a `.png`, `.pdf` or `.svg` file using the save button. These files can be opened with many image viewing programs and can be inserted into word-processed documents.

10.7 Changing properties of graphs

You can change the thickness of curves plotted in graphs by including an additional argument in the `wxplot2d` command, of the form

```
[style, [lines, 3]]
```

where the number indicates how thick the line should be. By default, lines have thickness 1.

So, to plot a graph of the equation $y = 2x$ with a thicker line, enter, for example,

```
wxplot2d(2*x, [x, 0, 1], [style, [lines, 3]]);
```

Typing this additional argument each time you want to plot a graph can be cumbersome, so you might like to assign this argument to a variable that you can reuse. For example, to assign your preferred style to the variable `thick` enter

```
thick:  [style, [lines, 3]];
```

Then you can plot graphs using commands such as

```
wxplot2d(2*x, [x, 0, 1], thick);
```

To change the colour of a curve, you can include an argument of the form `[color, red]` as described in Subsection 7.2. Alternatively, you can add a further element to the styles argument described above. For example, including

```
[style, [lines, 3, 5]]
```

produces curves with thickness 3 and colour number 5, which is black. The colour number codes used by the `style` argument are given in the table below.

`style` argument colour numbers

Number	Colour
1	blue
2	red
3	green
4	magenta
5	black
6	cyan

Note that the value of any variable you define to help set curve properties (such as `thick` above) will be lost when you close Maxima at the end of a session unless you save your session. The `style` argument also only changes properties of the curve plotted, not other aspects of the graph such as the axes and labels.

Changing the axes and labels, and ensuring such changes are automatically used each time Maxima is started, is more complicated. A computer file is provided in the Accessibility section of the OU Maxima website `learn1.open.ac.uk/site/maxima` which you can download and configure to set the graph properties to your needs. These settings will then be automatically used for each graph in every Maxima session. Details of how to download and use this are given in the Accessibility section of the OU Maxima website.

Solutions to Computer activities

Solution to Computer activity 3

```
(%i1) 2+3*4/5;
```

(%o1) $\frac{22}{5}$

So $2 + 3 \times \frac{4}{5} = \frac{22}{5}$.

Solution to Computer activity 5

(a)
```
(%i1) 32^(1/5);
```
(%o1) 2

So $\sqrt[5]{32} = 2$. Note that Maxima returns the real fifth root of 32.

(b)
```
(%i2) sin(%pi/3);
```
(%o2) $\frac{\sqrt{3}}{2}$

So $\sin\left(\frac{\pi}{3}\right) = \frac{\sqrt{3}}{2}$.

(c)
```
(%i3) tan(30*%pi/180);
```
(%o3) $\frac{1}{\sqrt{3}}$

So $\tan 30° = \frac{1}{\sqrt{3}}$.

(d)
```
(%i4) %e^4;
```
(%o4) $\%e^4$

(e)
```
(%i5) log(3.4);
```
(%o5) 1.223775431622116

So $\ln 3.4 \approx 1.223\,775\,431\,622\,116$.

Do not be surprised if your line numbering differs from ours. Each of our solutions is prepared as a separate Maxima session with the line numbering starting at 1 (unless it is a continuation of a previous activity); your numbering will continue to increase for the duration of your session.

Solution to Computer activity 6

(a)
```
(%i1) 1309/3451;
```
(%o1) $\frac{11}{29}$

(b)
```
(%i2) 1309.0/3451;
```
(%o2) 0.3793103448275862

So $1309.0/3451 \approx 0.379\,310\,344\,827\,5862$.

(c)
```
(%i3) float(1309/3451);
```
(%o3) 0.3793103448275862

So $1309/3451 \approx 0.379\,310\,344\,827\,5862$.

Your answers may differ in the last decimal place or so to those given here. This is due to small differences in the way floating-point numbers are computed by different systems.

Solution to Computer activity 7

(a)
```
(%i1) float(sqrt(325));
(%o1) 18.02775637731995
```
So $\sqrt{325} \approx 18.027\,756\,377\,319\,95$.

(b)
```
(%i2) log(10), numer;
(%o2) 2.302585092994046
```
So $\ln 10 \approx 2.302\,585\,092\,994\,046$.

Solution to Computer activity 9

(a)
```
(%i1) ? float;
-- Function: float (<expr>)
     Converts integers, rational numbers and bigfloats in
     <expr> to floating point numbers.  It is also an
     'evflag', 'float' causes non-integral rational numbers
     and bigfloat numbers to be converted to floating point.
  There are also some inexact matches for 'float'.
  Try '?? float' to see them.
(%o1) true
```

(b)
```
(%i2) ?? sqr;
 0: isqrt  (Functions and Variables for Number Theory)
 1: sqrt  (Root Exponential and Logarithmic Functions)
 2: sqrtdenest  (Package sqdnst)
 3: sqrtdispflag  (Functions and Variables for Display)
--> 1;
-- Function: sqrt (<x>)
     The square root of <x>.  It is represented internally by
     '<x>^(1/2)'.  See also 'rootscontract' and 'radexpand'.
(%o2) true
```

Solution to Computer activity 10

(a)
```
(%i1) a:x^2+3;
```
(%o1) $x^2 + 3$

(b)
```
(%i2) a;
```
(%o2) $x^2 + 3$

(c)
```
(%i3) b:a^2;
```
(%o3) $(x^2 + 3)^2$

(d)
```
(%i4) print(a,b);
```
$x^2 + 3\ (x^2 + 3)^2$

(%o4) $(x^2 + 3)^2$

(e)
```
(%i5) a:x^3+y;
```
(%o5) $y + x^3$

(f)
```
(%i6) b;
```
(%o6) $(x^2 + 3)^2$

(g)
```
(%i7) values;
```
(%o7) [a, b]

(h)
```
(%i8) kill(b);
```
(%o8) *done*

(i)
```
(%i9) b;
```
(%o9) b

(j)
```
(%i10) a;
```
(%o10) $y + x^3$

(k)
```
(%i11) subst(3,x,a);
```
(%o11) $y + 27$

So when $x = 3$, $y + x^3 = y + 27$.

(l)
```
(%i12) ev(a,x=2);
```
(%o12) $y + 8$

So when $x = 2$, $y + x^3 = y + 8$.

(m)
```
(%i13) ev(a,[x=2,y=1]);
```
(%o13) 9

So when $x = 2$ and $y = 1$, $y + x^3 = 9$.

Solution to Computer activity 11

(a)
```
(%i1) f(x):=(x^4+x+1)/(x^3-13*x+12);
```
(%o1) $f(x) := \frac{x^4+x+1}{x^3-13x+12}$

(b)
```
(%i2) f(2);
```
(%o2) $-\frac{19}{6}$

So $f(2) = -\frac{19}{6}$.

(c)
```
(%i3) f(3);
```
```
expt: undefined: 0 to a negative exponent.
#0: f(x=3)
-- an error. To debug this try: debugmode(true);
```

(d)
```
(%i4) g(1);
```
(%o4) g(1)

(e)
```
(%i5) functions;
```
(%o5) [f(x)]

Solution to Computer activity 12

(a)
```
(%i1) h(x,y):=x*y^2;
```
(%o1) $h(x, y) := x\,y^2$

(b)
```
(%i2) h(2,3);
```
(%o2) 18

So $h(2, 3) = 18$.

Solution to Computer activity 13

(a)
```
(%i1) A: [2,3,5,7,11];
```
(%o1) $[2, 3, 5, 7, 11]$

(b)
```
(%i2) A[4];
```
(%o2) 7

So the fourth element of `A` is 7.

(c)
```
(%i3) third(A);
```
(%o3) 5

So the third element of `A` is 5.

(d)
```
(%i4) last(A);
```
(%o4) 11

So the last element of `A` is 11.

(e)
```
(%i5) length(A);
```
(%o5) 5

So the list `A` has 5 elements, as expected.

(f)
```
(%i6) B:append(A, [13,17]);
```
(%o6) $[2, 3, 5, 7, 11, 13, 17]$
```
(%i7) length(B);
```
(%o7) 7

So `B` contains 7 elements, as expected.

Solution to Computer activity 14

(a)
```
(%i1) a:makelist(i^2, i, 1, 10);
```
(%o1) $[1, 4, 9, 16, 25, 36, 49, 64, 81, 100]$

(b)
```
(%i2) makelist(i^3, i, 1, 19, 2);
```
(%o2) $[1, 27, 125, 343, 729, 1331, 2197, 3375, 4913, 6859]$

(c)
```
(%i3) makelist(i+1, i, a);
```
(%o3) $[2, 5, 10, 17, 26, 37, 50, 65, 82, 101]$

(d)
```
(%i4) create_list(2^i*3^j, i, [1,2,3], j, [2,4,6]);
```
(%o4) $[18, 162, 1458, 36, 324, 2916, 72, 648, 5832]$

(e)
```
(%i5) create_list(x^i*y^j, i,1,3, j,1,2);
```
(%o5) $[x\,y, x\,y^2, x^2\,y, x^2\,y^2, x^3\,y, x^3\,y^2]$

Solution to Computer activity 15

(a)
```
(%i1) b:makelist([i,i^2], i, 1, 9);
```
(%o1) $[[1, 1], [2, 4], [3, 9], [4, 16], [5, 25], [6, 36], [7, 49], [8, 64], [9, 81]]$

(b)
```
(%i2) b[6];
```
(%o2) $[6, 36]$

So the sixth element of `b` is the list `[6,36]`.

(c)
```
(%i3) b[6][2];
(%o3) 36
```

So the second element of the list that is the sixth element of `b` is 36.

Solution to Computer activity 16

(a)
```
(%i1) A: [2,3,5,7,11];
(%o1) [2,3,5,7,11]
(%i2) sum(A[n],n,1,4);
(%o2) 17
```

So the sum of the first four prime numbers is 17.

(b)
```
(%i3) sum(i^2,i,1,100);
(%o3) 338350
```

So $\sum_{i=1}^{100} i^2 = 338\,350$.

Solution to Computer activity 17

(a)
```
(%i1) v:transpose([2,3,6]);
```
$$(\%o1)\quad \begin{bmatrix} 2 \\ 3 \\ 6 \end{bmatrix}$$

(b)
```
(%i2) w:transpose([1,-2,4]);
```
$$(\%o2)\quad \begin{bmatrix} 1 \\ -2 \\ 4 \end{bmatrix}$$

(c)
```
(%i3) 2*v;
```
$$(\%o3)\quad \begin{bmatrix} 4 \\ 6 \\ 12 \end{bmatrix}$$

So $2\mathbf{v} = \begin{bmatrix} 4 \\ 6 \\ 12 \end{bmatrix}$.

(d)
```
(%i4) 3*v+2*w;
```
$$(\%o4)\quad \begin{bmatrix} 8 \\ 5 \\ 26 \end{bmatrix}$$

So $3\mathbf{v} + 2\mathbf{w} = \begin{bmatrix} 8 \\ 5 \\ 26 \end{bmatrix}$.

(e)
```
(%i5) v.w;
(%o5) 20
```

So $\mathbf{v} \cdot \mathbf{w} = 20$.

(f)
```
(%i6) sqrt(v.v);
(%o6) 7
```

So $|\mathbf{v}| = \sqrt{\mathbf{v} \cdot \mathbf{v}} = 7$.

Solution to Computer activity 18

(a)
```
(%i1) P:matrix( [2,-3,3], [-2,1,2], [1,-1,4] );
```
$$\text{(\%o1)} \quad \begin{bmatrix} 2 & -3 & 3 \\ -2 & 1 & 2 \\ 1 & -1 & 4 \end{bmatrix}$$

(b)
```
(%i2) Q:matrix( [5,-3], [-1,-2], [0,4] );
```
$$\text{(\%o2)} \quad \begin{bmatrix} 5 & -3 \\ -1 & -2 \\ 0 & 4 \end{bmatrix}$$

(c)
```
(%i3) R:matrix( [1,-2,-1], [4,7,0], [-5,1,3] );
```
$$\text{(\%o3)} \quad \begin{bmatrix} 1 & -2 & -1 \\ 4 & 7 & 0 \\ -5 & 1 & 3 \end{bmatrix}$$

Solution to Computer activity 19

(a)
```
(%i4) P+R;
```
$$\text{(\%o4)} \quad \begin{bmatrix} 3 & -5 & 2 \\ 2 & 8 & 2 \\ -4 & 0 & 7 \end{bmatrix}$$

So $\mathbf{P}+\mathbf{R} = \begin{bmatrix} 3 & -5 & 2 \\ 2 & 8 & 2 \\ -4 & 0 & 7 \end{bmatrix}$.

(b)
```
(%i5) 3*P;
```
$$\text{(\%o5)} \quad \begin{bmatrix} 6 & -9 & 9 \\ -6 & 3 & 6 \\ 3 & -3 & 12 \end{bmatrix}$$

So $3\mathbf{P} = \begin{bmatrix} 6 & -9 & 9 \\ -6 & 3 & 6 \\ 3 & -3 & 12 \end{bmatrix}$.

(c)
```
(%i6) R.Q;
```
$$\text{(\%o6)} \quad \begin{bmatrix} 7 & -3 \\ 13 & -26 \\ -26 & 25 \end{bmatrix}$$

So $\mathbf{RQ} = \begin{bmatrix} 7 & -3 \\ 13 & -26 \\ -26 & 25 \end{bmatrix}$.

(d)
```
(%i7) P^^2;
```
$$\text{(\%o7)} \quad \begin{bmatrix} 13 & -12 & 12 \\ -4 & 5 & 4 \\ 8 & -8 & 17 \end{bmatrix}$$

So $\mathbf{P}^2 = \begin{bmatrix} 13 & -12 & 12 \\ -4 & 5 & 4 \\ 8 & -8 & 17 \end{bmatrix}$.

(e)
```
(%i8) determinant(P);
```
(%o8) -15

So $\det \mathbf{P} = -15$.

(f)
```
(%i9) invert(R);
```
(%o9)
$$\begin{bmatrix} \frac{7}{2} & \frac{5}{6} & \frac{7}{6} \\ -2 & -\frac{1}{3} & -\frac{2}{3} \\ \frac{13}{2} & \frac{3}{2} & \frac{5}{2} \end{bmatrix}$$

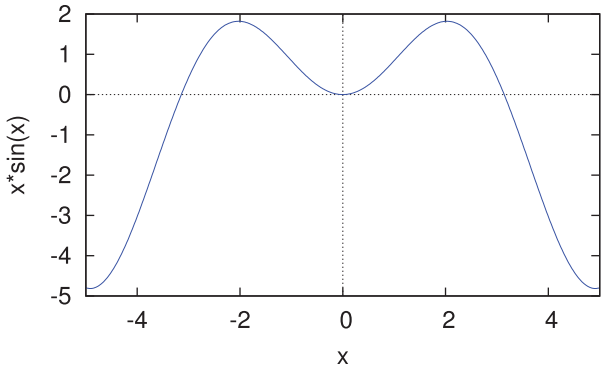
So $\mathbf{R}^{-1} = \begin{bmatrix} \frac{7}{2} & \frac{5}{6} & \frac{7}{6} \\ -2 & -\frac{1}{3} & -\frac{2}{3} \\ \frac{13}{2} & \frac{3}{2} & \frac{5}{2} \end{bmatrix}$.

(g)
```
(%i10) x:matrix([0],[1],[1]);
```
(%o10)
$$\begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$$
```
(%i11) P.x;
```
(%o11)
$$\begin{bmatrix} 0 \\ 3 \\ 3 \end{bmatrix}$$

Since $\mathbf{Px} = \begin{bmatrix} 0 \\ 3 \\ 3 \end{bmatrix} = 3\mathbf{x}$, where $\mathbf{x} = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$, it follows that $\mathbf{x}$ is an eigenvector of $\mathbf{P}$ and the corresponding eigenvalue is 3.

Solution to Computer activity 20

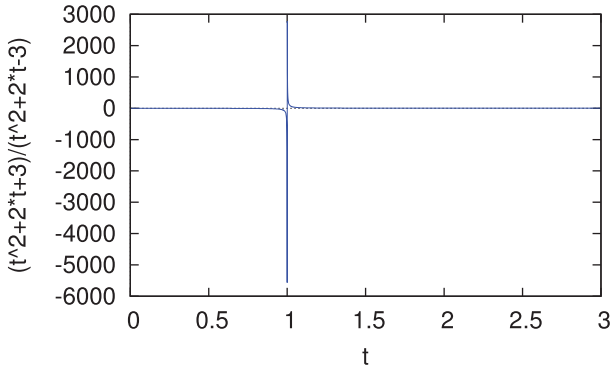
(a)
```
(%i1) wxplot2d(x*sin(x), [x,-5,5]);
```
(%t1)



(%o1)

(b)
```
(%i2) f(t):=(t^2+2*t+3)/(t^2+2*t-3);
```
(%o2) $\mathtt{f(t)} := \dfrac{\mathtt{t}^2 + 2\mathtt{t} + 3}{\mathtt{t}^2 + 2\mathtt{t} - 3}$
```
(%i3) wxplot2d(f(t), [t,0,3]);
```

(%t3)

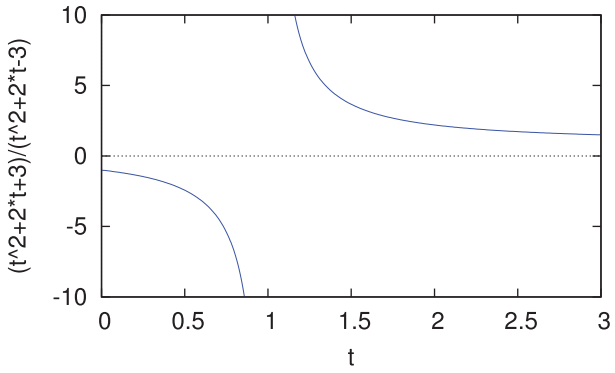


(%o3)

(c) (%i4) `wxplot2d(f(t), [t,0,3], [y,-10,10]);`

```
plot2d:  some values were clipped.
```

(%t4)



(%o4)

Solution to Computer activity 21

(a) (%i1) `p(x):=x*cos(x);`

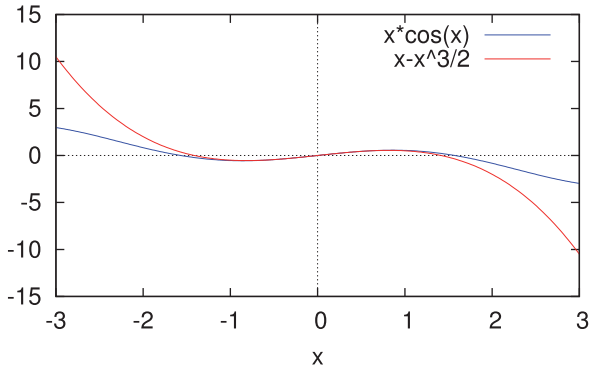
(%o1) $p(x) := x\cos(x)$

(%i2) `q(x):=x-x^3/2;`

(%o2) $q(x) := x - \frac{x^3}{2}$

(b) (%i3) `wxplot2d([p(x),q(x)],[x,-3,3]);`

(%t3)

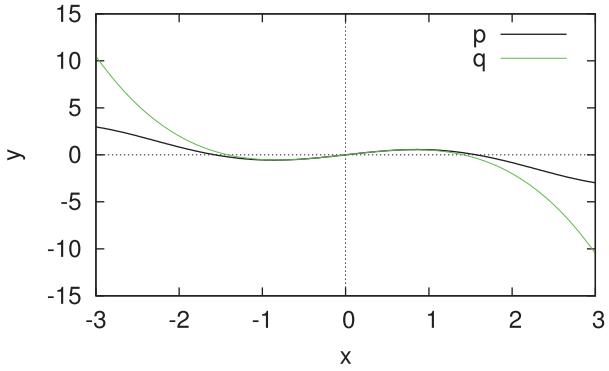


(%o3)

Solution to Computer activity 22

```
(%i4) wxplot2d([p(x),q(x)], [x,-3,3], [color, black, green],
       [legend, "p", "q"], [ylabel, "y"]);
```

(%t4)

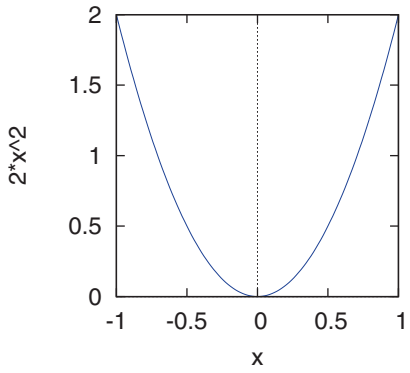


(%o4)

Solution to Computer activity 23

```
(%i1) wxplot2d(2*x^2,[x,-1,1], same_xy);
```

(%t1)



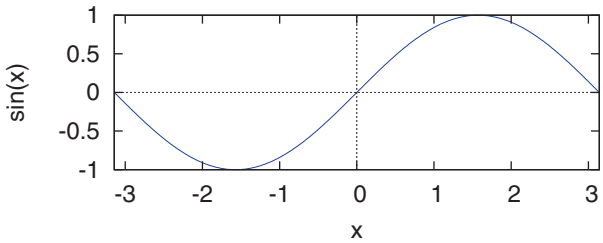
(%o1)

Solution to Computer activity 24

(a)
```
(%i1) set_plot_option(same_xy)$
```

(b)
```
(%i2) wxplot2d(sin(x),[x,-%pi,%pi]);
```

(%t2)

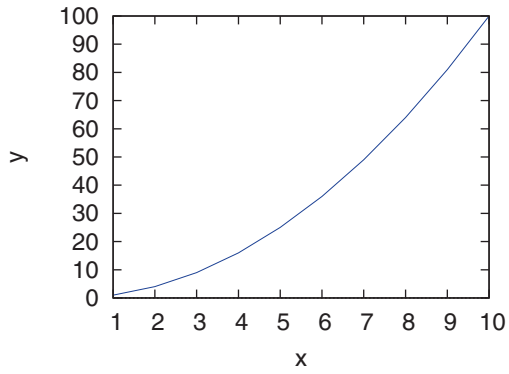


(%o2)

Solution to Computer activity 25

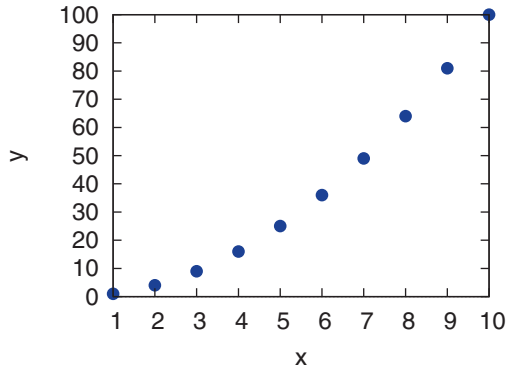
(a)
```
(%i1) pts:makelist([i,i^2], i, 1, 10);
```
(%o1) $[[1,1],[2,4],[3,9],[4,16],[5,25],[6,36],[7,49],[8,64],[9,81],[10,100]]$

(b)
```
(%i2) wxplot2d([discrete, pts]);
```
(%t2)



(%o2)

(c)
```
(%i3) wxplot2d([discrete, pts], [style, points]);
```
(%t3)



(%o3)

Solution to Computer activity 26

(a)
```
(%i1) sol:solve(x^3-4*x^2+2*x+4=0);
```
(%o1) $[x = 1-\sqrt{3}, x = \sqrt{3}+1, x = 2]$

So the three solutions are $x = 1-\sqrt{3}$, $x = 1+\sqrt{3}$ and $x = 2$.

(b)
```
(%i2) solve(y=log(u*x+v), x);
```
(%o2) $[x = \frac{\%e^{y}-v}{u}]$

So the rearranged equation is $x = \dfrac{e^y - v}{u}$.

(c)
```
(%i3) solve(tan(3*x-1)=sqrt(3));
```
```
solve:  using arc-trig functions to get a solution.
Some solutions will be lost.
```
(%o3) $[x = \frac{\pi+3}{9}]$

So one solution of the equation is $x = \dfrac{\pi+3}{9}$.

(d) `(%i4) solve([y=x^2-6*x+9, (x-3)^2+(y-2)^2=3], [x,y]);`

`(%o4)` $[[\mathtt{x}=-\frac{\sqrt{5}-5}{2},\mathtt{y}=\frac{\sqrt{5}+3}{2}],[\mathtt{x}=\frac{\sqrt{5}+5}{2},\mathtt{y}=-\frac{\sqrt{5}-3}{2}],$
$[\mathtt{x}=-\frac{\sqrt{5}-7}{2},\mathtt{y}=-\frac{\sqrt{5}-3}{2}],[\mathtt{x}=\frac{\sqrt{5}+7}{2},\mathtt{y}=\frac{\sqrt{5}+3}{2}]]$

So the solutions are $x=\frac{5-\sqrt{5}}{2}$, $y=\frac{3+\sqrt{5}}{2}$; $x=\frac{5+\sqrt{5}}{2}$, $y=\frac{3-\sqrt{5}}{2}$; $x=\frac{7-\sqrt{5}}{2}$, $y=\frac{3-\sqrt{5}}{2}$ and $x=\frac{7+\sqrt{5}}{2}$, $y=\frac{3+\sqrt{5}}{2}$.

(e) `(%i5) solve(x^9+2*x-1=0);`

`(%o5)` $[0=\mathtt{x}^9+2\mathtt{x}-1]$

Maxima was unable to solve this equation.

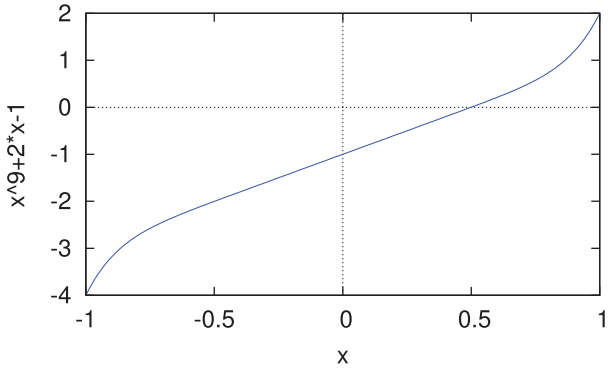
Solution to Computer activity 27

(a) `(%i1) f(x):=x^9+2*x-1;`

`(%o1)` $\mathtt{f(x)}:=\mathtt{x}^9+2\mathtt{x}-1$

(b) `(%i2) wxplot2d(f(x), [x,-1,1]);`

`(%t2)`



`(%o2)`

Since `f(0)` <0 and `f(1)` >0, there must be a solution in the interval $(0,1)$.

(c) `(%i3) find_root(f(x), x, 0, 1);`

`(%o3)` 0.4990401803569676

So a solution of $x^9+2x-1=0$ is 0.499 (to 3 s.f.).

Solution to Computer activity 28

`(%i1) allroots(x^9+2*x-1=0);`

`(%o1)` $[\mathtt{x}=0.4990401803569676,$
$\mathtt{x}=0.4207825521114528\%\mathtt{i}-1.058521241857364,$
$\mathtt{x}=-0.4207825521114528\%\mathtt{i}-1.058521241857364,$
$\mathtt{x}=1.01795501023027\%\mathtt{i}-0.4714669586482951,$
$\mathtt{x}=-1.01795501023027\%\mathtt{i}-0.4714669586482951,$
$\mathtt{x}=1.024451854329254\%\mathtt{i}+0.3548245783739534,$
$\mathtt{x}=0.3548245783739534-1.024451854329254\%\mathtt{i},$
$\mathtt{x}=0.4326584474822993\%\mathtt{i}+0.9256435319532224,$
$\mathtt{x}=0.9256435319532224-0.4326584474822993\%\mathtt{i}]$

So there is one real root, $x = 0.499$ (to 3 s.f.), and eight complex roots, occurring in conjugate pairs.

Solution to Computer activity 29

(a)
```
(%i1) diff(cos(x)/(x^2+3), x);
```
(%o1) $-\dfrac{\sin(x)}{x^2+3} - \dfrac{2\,x\cos(x)}{(x^2+3)^2}$

So the derivative of $f(x) = \dfrac{\cos x}{x^2+3}$ is $f'(x) = -\dfrac{\sin x}{x^2+3} - \dfrac{2x\cos x}{(x^2+3)^2}$.

(b)
```
(%i2) diff(u^2*sin(u), u, 2);
```
(%o2) $-u^2\sin(u) + 2\sin(u) + 4\,u\cos(u)$

So the second derivative of $g(u) = u^2 \sin u$ is $g''(u) = -u^2 \sin u + 2\sin u + 4u\cos u$.

(c)
```
(%i3) diff(log(t)/t^2, t, 3);
```
(%o3) $\dfrac{26}{t^5} - \dfrac{24\log(t)}{t^5}$

So the third derivative of $h(t) = \dfrac{\ln t}{t^2}$ is $h'''(t) = \dfrac{26}{t^5} - \dfrac{24\ln t}{t^5}$.

(d)
```
(%i4) integrate(x*sin(x)^2, x);
```
(%o4) $-\dfrac{2\,x\sin(2\,x) + \cos(2\,x) - 2\,x^2}{8}$

So $\displaystyle\int x\sin^2 x\,dx = -\frac{2x\sin(2x) + \cos(2x) - 2x^2}{8} + c$.

(e)
```
(%i5) integrate(t^2*%e^t, t, 0, 1);
```
(%o5) $\%e - 2$

So $\displaystyle\int_0^1 t^2 e^t\,dt = e - 2$.

(f)
```
(%i6) integrate(1/x^2, x, 1, 2);
```
(%o6) $\dfrac{1}{2}$

So $\displaystyle\int_1^2 \frac{1}{x^2}\,dx = \frac{1}{2}$.

(g)
```
(%i7) integrate(tan(exp(-x)), x, 0, 1);
```
(%o7) $\displaystyle\int_0^1 \tan(\%e^{-x})dx$

Maxima cannot evaluate $\displaystyle\int_0^1 \tan(e^{-x})\,dx$ exactly.

Solution to Computer activity 30

(a)
```
(%i1) f(x):=x*sqrt(1+4*x^2);
```
(%o1) $f(x) := x\sqrt{1+4\,x^2}$

(b)
```
(%i2) df(x):=diff(f(x),x);
```
(%o2) $df(x) := \dfrac{d}{dx}f(x)$

(c)
```
(%i3) df(3);
diff:  second argument must be a variable; found 3
#0:  df(x=3)
-- an error.  To debug this try:  debugmode(true);
```

(d)
```
(%i4) df(x):=''(diff(f(x),x));
```
(%o4) $\mathtt{df(x)} := \sqrt{4\,\mathtt{x}^2+1} + \dfrac{4\,\mathtt{x}^2}{\sqrt{4\,\mathtt{x}^2+1}}$

So $f'(x) = \sqrt{1+4x^2} + \dfrac{4x^2}{\sqrt{1+4x^2}}$.

(e)
```
(%i5) df(3);
```
(%o5) $\dfrac{73}{\sqrt{37}}$

So $f'(3) = \dfrac{73}{\sqrt{37}}$.

(f)
```
(%i6) F(x):=''(integrate(f(x),x));
```
(%o6) $\mathtt{F(x)} := \dfrac{(4\,\mathtt{x}^2+1)^{3/2}}{12}$

(g)
```
(%i7) F(1)-F(0);
```
(%o7) $\dfrac{5^{3/2}}{12} - \dfrac{1}{12}$
```
(%i8) integrate(f(x),x,0,1)
```
(%o8) $\dfrac{5^{3/2}}{12} - \dfrac{1}{12}$

So $\int_0^1 f(x)\,dx = F(1) - F(0) = \dfrac{5^{3/2}}{12} - \dfrac{1}{12}$.

Solution to Computer activity 31

(a) (i)
```
(%i1) kill(n);
(%o1) done
```

(ii)
```
(%i2) integrate(x^n, x);
Is n equal to -1?  n;
```
(%o2) $\dfrac{\mathtt{x}^{\mathtt{n}+1}}{\mathtt{n}+1}$

(b) (i)
```
(%i3) assume(notequal(n,-1));
```
(%o3) [notequal(n, −1)]

(ii)
```
(%i4) integrate(x^n, x);
```
(%o4) $\dfrac{\mathtt{x}^{\mathtt{n}+1}}{\mathtt{n}+1}$

(iii)
```
(%i5) facts();
```
(%o5) [notequal(n, −1)]

So Maxima knows $n \neq -1$.

(iv)
```
(%i6) forget(notequal(n,-1));
```
(%o6) [notequal(n, −1)]

(v)
```
(%i7) facts();
```
(%o7) []

No facts are now known.

Maxima reference guide

Mathematical operations and functions

Operation or function	Syntax	Example
Addition	+	4+3;
Subtraction	-	4-3;
Multiplication	*	4*3;
Division	/	4/3;
Brackets	(*and*)	2*(3+4);
Powers, for example 2^3	^	2^3;
	or **	2**3;
Square root, for example $\sqrt{5}$	sqrt( )	sqrt(5);
Exponential, for example e^3	%e^	%e^3;
	or exp( )	exp(3);
Natural logarithm, ln	log( )	log(8);
Magnitude (absolute value)	abs( )	abs(-3);
sin	sin( )	sin(1);
cos	cos( )	cos(3*%pi/2);
tan	tan( )	tan(%pi/4);
cosec	csc( )	csc(2);
sec	sec( )	sec(%pi/3);
cot	cot( )	cot(%pi/4);
arcsin	asin( )	asin(sqrt(3)/2);
arccos	acos( )	acos(1/sqrt(2));
arctan	atan( )	atan(1/2);
sinh	sinh( )	sinh(1);
cosh	cosh( )	cosh(1);
tanh	tanh( )	tanh(1);
cosech	csch( )	csch(1);
sech	sech( )	sech(0);
coth	coth( )	coth(1);
arcsinh	asinh( )	asinh(1);
arccosh	acosh( )	acosh(1);
arctanh	atanh( )	atanh(1/2);
Equality (within an equation)	=	5=2*x+3;
Vector or matrix addition	+	P+Q;
Vector or matrix subtraction	-	P-Q;
Scalar multiple of a vector or matrix	*	2*P;
Vector scalar (dot) product or matrix multiplication	.	P.Q;
Matrix powers	^^	P^^3;
Transpose (of a vector or matrix)	transpose( )	transpose(P);
Determinant (of a square matrix)	determinant(matrix)	determinant(P);
Matrix inverse (of a square matrix)	invert(matrix)	invert(P);
	or matrix^^(-1)	P^^(-1);
Identity matrix	ident(size)	ident(2);

Mathematical commands

Operation	Command	Example
Get help on a command	`? command` *or* `describe(command)`	`? float;` `describe(float);`
Find help information whose title contains the given text	`??` *or* `describe(, inexact)`	`?? flo;` `describe(flo, inexact);`
Force the evaluation of a command	`''()`	`g(x):=''(diff(x^4,x));`
Assign a value to a variable	`variable:value`	`a:23;`
Display the value of a variable	`variable` `print(variable)`	`a;` `print(a);`
Display the value of several variables	`print(variable,variable,...)`	`print(a,b);`
Define a function	`:=`	`f(x):=2*x+3;` `f(x,y):=x*y;`
Evaluate a function at a value	`function()`	`f(1);`
Remove an assigned variable or function	`kill(variable)` `kill(function)`	`kill(a);` `kill(f);`
Remove all assigned variables and functions	`kill(all)`	`kill(all);`
Convert to a decimal number	`float()` *or* `,numer`	`float(sqrt(2));` `sqrt(2),numer;`
Substitute	`subst(value,variable,expression)`	`subst(4,x,x^2+1);` which substitutes 4 for x in x^2+1
	`ev(expression,variable=value)`	`ev(x^2+1,x=4);` which substitutes 4 for x in x^2+1
	`ev(expression,[variable=value, variable=value,...])`	`ev(x^2+y,[x=1,y=2]);` which substitutes 1 for x and 2 for y in x^2+y
Left-hand side of an equation	`lhs(equation)`	`lhs(4*x+1=2*x-2);`
Right-hand side of an equation	`rhs(equation)`	`rhs(4*x+1=2*x-2);`
Solve an equation exactly	`solve(equation)`	`solve(2*x^2-1=0);`
Solve an equation for a variable, i.e. change the subject	`solve(equation, variable)`	`solve(2*a*b-3*b=0, a);`
Solve simultaneous equations	`solve(list of equations, list of variables)`	`solve([2*x+y=4, x-2*y=1], [x,y]);`
Find (numerically) a root of an equation within a given interval Solve an equation numerically	`find_root(expression,variable, interval start value, interval end value)`	`find_root(2*x^3-1,x,0,1);`
Find (numerically) all roots of a polynomial equation	`allroots(equation)`	`allroots(2*x^3-1=0);`

Mathematical commands (continued)

Operation	Command	Example
Form a list or vector	`[ , ,...]`	`[a,b,c];`
Assign a list or vector to a variable	`variable:[ , ,...]`	`A:[a,b,c];`
Create a list	`makelist( general term , dummy variable , start value , end value )`	`makelist(2*n,n,1,100);`
	`makelist( general term , dummy variable , start value , end value , step )`	`makelist(2*n,n,1,100,2);`
	`makelist( general term , dummy variable , list )`	`makelist(2*n,n,A);`
	`create_list( general term , dummy variable 1 , list , dummy variable 2 , list , ...)`	`create_list(x^i*y^j, i,[1,2,3], j,[2,4,6]);`
	`create_list( general term , dummy variable 1 , start , end , dummy variable 2 , start , end , ...)`	`create_list(x^i*y^j, i,1,3, j,1,2);`
Extract an element of a list	`list [ index ]`	`A[2];`
Extract the first, second, ..., tenth element of a list	`first( list ), second( list ), ..., tenth( list )`	`first(A);`
Extract the last element of a list	`last( list )`	`last(A);`
Combine lists	`append( list , list ,...)`	`append([1,2],[3,4]);`
Find the length of a list	`length( list )`	`length(A);`
Sum the elements of a list or terms of a series	`sum( general term , dummy variable , start value , end value )`	`sum(i^2,i,1,100);`
Specify a matrix	`matrix( row , row ,...)`	`matrix([1,2],[3,4]);`
Differentiate	`diff( , variable )`	`diff(sin(x^2),x);`
Differentiate multiple times	`diff( , variable , positive integer )`	`diff(log(x),x,3);` to differentiate three times
Integrate (find an antiderivative)	`integrate( , variable )`	`integrate(x^2,x);`
Evaluate a definite integral	`integrate( , variable , lower limit , upper limit )`	`integrate(sin(x),x,0,1);`
Make an assumption about a variable	`assume( )`	`assume(a>0);`
State two things are equal	`equal( , )`	`assume(equal(n,3));`
State two things are not equal	`notequal( , )`	`assume(notequal(n,-1));`
Forget a property	`forget( property )`	`forget(a>0);`
List all known facts	`facts()`	`facts();`
List all known facts about a particular variable	`facts( variable )`	`facts(a);`

Mathematical commands (continued)

Operation	Command	Example
Plot a graph	`wxplot2d(expression, horizontal range,...)`	`wxplot2d(x^2,[x,0,1]);` which plots the graph of x^2 for x between 0 and 1 `wxplot2d(x^2,[x,0,1],[y,0,2]);` which plots the graph of x^2 for x between 0 and 1, and with vertical axis between 0 and 2
Plot several graphs	`wxplot2d(list of expressions, horizontal range,...)`	`wxplot2d([x^2,2*x],[x,0,1]);` which plots graphs of x^2 and $2x$ for x between 0 and 1
Plot a set of points	`wxplot2d([discrete, list of coordinate pairs], [style, points], ...)` (use `plot2d` similarly if not using wxMaxima)	`wxplot2d([discrete, [[1,5], [2,10]]], [style, points]);`
Set the default plotting options	`set_plot_option(option)`	`set_plot_option(same_xy);`

Mathematical constants

Constant	Syntax	Example
e	`%e`	`%e^2;`
π	`%pi`	`2*%pi;`
i	`%i`	`2+3*%i;`

Maxima system variables

Variable	Description	Example
`fpprintprec`	The number of significant figures of a decimal number to display	`fpprintprec:3;`
`functions`	The list of all user-defined functions	`functions;`
`values`	The list of all user-assigned variables	`values;`

Maxima graph plotting options

The following options can be added as additional arguments to plotting commands.

Option	Argument	Example
Vertical range	`[y, , ]`	`[y, 0, 5]`
Curve colour	`[color, , ...]`	`[color, red]`
Legend text	`[legend, " ", ...]`	`[legend, "Car A"]`
Turn legend off	`[legend, false]`	`[legend, false]`
Position legend at the bottom of the graph	`[gnuplot_preamble, "set key bottom"]`	`[gnuplot_preamble, "set key bottom"]`
Position legend to the right of the graph	`[gnuplot_preamble, "set key outside"]`	`[gnuplot_preamble, "set key outside"]`
Horizontal axis label	`[xlabel, " "]`	`[xlabel, "t (h)"]`
Vertical axis label	`[ylabel, " "]`	`[ylabel, "s (km)"]`
Set axes to have equal scales	`[same_xy, true]`	`[same_xy, true]`
Change the size of the graph	`wxplot_size:[ , ]`	`wxplot_size:[800,600];`
Plot points	`[style, points]`	`[style, points]`
Plot dots	`[style, dots]`	`[style, dots]`
Plot a line and points	`[style, linespoints]`	`[style, linespoints]`

wxMaxima keyboard sequences (Microsoft Windows)

Function	Key sequence
Zoom in	`Alt-I`
Zoom out	`Alt-O`
Copy	`Ctrl-C`
Interrupt a calculation	`Ctrl-G`
Re-evaluate the worksheet	`Ctrl-R`
Close wxMaxima	`Ctrl-Q`
Paste	`Ctrl-V`

wxMaxima menu commands (Microsoft Windows)

Operation	Menu name	Menu option
Close wxMaxima	`File`	`Exit`
Save your work	`File`	`Save` or `Save As`
Print the worksheet	`File`	`Print`
Open a worksheet	`File`	`Open`
Configure wxMaxima	`Edit`	`Configure`
Copy	`Edit`	`Copy`
Paste	`Edit`	`Paste`
Zoom in	`View`	`Zoom In`
Zoom out	`View`	`Zoom Out`
Toggle display of main toolbar	`View`	`Main Toolbar`
Re-evaluate worksheet	`Cell`	`Evaluate All Visible Cells`
Insert text cell	`Cell`	`Insert Text cell`
Insert title cell	`Cell`	`Insert Title cell`
Insert section cell	`Cell`	`Insert Section cell`
Insert subsection cell	`Cell`	`Insert Subsection cell`
Interrupt a calculation	`Maxima`	`Interrupt`
Restart Maxima	`Maxima`	`Restart Maxima`
Enter a matrix	`Algebra`	`Enter Matrix...`
Maxima help window	`Help`	`Maxima Help`

Maxima command-line interface commands

Operation	Command
Close Maxima	`quit();`
Interrupt a calculation	`Ctrl-C`
Turn off 2D formatting of maths	`display2d:false;`
Display output left justified	`leftjust:true;`
Start writing a transcript	`writefile("file path");`
Stop writing a transcript	`closefile();`
Playback all the input and output in a session	`playback();`
Playback a range of input and output	`playback([start line number, end line number]);`
Save the state of a session	`save("file path", all);`
Reload a session	`load("file path");`

Acknowledgements

Grateful acknowledgement is made to the following source:

Page 4: Taken from: http://www.ma.utexas.edu/people/in_memoriam/ (Accessed 2014)

Every effort has been made to contact copyright holders. If any have been inadvertently overlooked the publishers will be pleased to make the necessary arrangements at the first opportunity.

Index

abort calculation
  using the command-line interface 55
  using wxMaxima 17
`abs` 13
absolute value 13
accessibility 50
  changing wxMaxima colours and fonts 51
  using a screen magnifier 52
  using a screenreader 52
  using wxMaxima with keyboard alone 51
`acos` 13
addition 13
  of matrices 34
  of vectors 31
algebraic expressions 22
`allroots` 44
analytic solution of equations 43
`append` 27
append elements to a list 27
arccos 13
arcsin 13
arctan 13
argument of a command 12
`asin` 13
assign
  a derivative or integral to a function 48
  function 25
  variable 22
`assume` 49, 50
assumption
  about a variable 49, 50
  equals 50
  forget 50
  list all 50
  not equals 49, 50
`atan` 13
axis scales, on graph 39

brackets 13

CAS *see* computer algebra system
cell, in wxMaxima
  marker 9
changing the size of a graph, in wxMaxima 40
`closefile` 58
closing
  command-line interface 55
  wxMaxima 12
colour of graph 38
command
  argument of 12
  editing
    using the command-line interface 56
  force evaluation of 48
  reusing
    using the command-line interface 56
command-line interface 6, 52
  abort calculation 55
  closing 55
  creating transcript 57
  cursor, editing 53
  editing commands 56
  editing cursor 53
  graphs 60
  input line number 52
  interrupt calculation 55
  output line number 53
  plotting graphs 60
  printing 57
  reusing commands 56
  saving work 57
  transcript 57
computer algebra system 4
configure wxMaxima 11, 15
`cos` 13
cosec 13
`cot` 13
create a list
  using `create_list` 28
  using `makelist` 28
`create_list` 28
`csc` 13
cursor
  editing
    in command-line interface 53

decimal approximation of a number 16
define
  functions 25
  variables 22
definite integral 46
delete
  function 26
  variable 24
derivative 46
  assign to a function 48
`describe` 21
`determinant` 34
determinant of a matrix 34
`diff` 46
differentiate 46

`discrete` 41
discrete data
    plotting 41
division 13
`dots` 42

editing commands
    using the command-line interface 56
editing cursor
    using the command-line interface 53
element
    index of 27
    of a list 27
`equal` 50
equals
    assumption 50
equation
    changing the subject of 44
    left-hand side 44
    rearranging 44
    right-hand side 43, 44
    simultaneous, solving 43
    solving
        analytically 43
        numerically 44
        simultaneously 43
`erf` 47
`ev` 24
evaluate
    an expression 10
    an expression using the command-line interface 53
    force command to 48
`exp` 13
exponentials 13
expressions
    algebraic 22
    evaluating 10

fact *see* assumption
`facts` 50
file types, wxMaxima 18
`find_root` 44
`first` 27
`float` 16
floating-point number 16
force evaluation of a command 48
`forget` 50
forget assumption 50
front-ends 6
`functions` 26
functions 25
    assign 25
    define 25
    delete 26
    inverse trigonometric 13
    plotting 36
    trigonometric 13

`gamma_incomplete` 47
gnuplot 39
graphs 35
    axis scales 39
    changing default behaviour 40
    changing line thickness 60
    changing plot style 42
    changing properties 38, 60
    colours 38
    legend 37
    of discrete data 41
    of functions 36
    size of, in wxMaxima 40
    using command-line interface 60
    vertical range 36

help 20

indefinite integral 46
index (of a list element) 27
input line number 10
input prompt, wxMaxima 9
inputting
    list 27
    matrix 32
    vector 31
installing Maxima 7
integral
    assign to a function 48
    definite 46
    indefinite 46
`integrate` 46
interfaces 6
    command-line 6, 52
    wxMaxima 6, 8
interrupt calculation
    using the command-line interface 55
    using wxMaxima 17
inverse of a matrix 34
inverse trigonometric functions 13
`invert` 34

`kill`
    function 26
    variable 24

`last` 27
left-hand side, of an equation 44
legend, of a graph 37
  position 39
  suppress 39
`length` 27
length of a list 27
`lhs` 44
line number
  input 10
  output 10
`linespoints` 42
list 27
  append elements 27
  create using `create_list` 28
  create using `makelist` 28
  element of 27
  index 27
  inputting 27
  length of 27
  of assumptions 50
  of lists 29
  summing 30
`load` 59
  previous session
    using the command-line interface 59
    using wxMaxima 18
`log` 13
logarithms
  natural 13
  other bases 13

magnitude 13
`makelist` 28
mathematical operations 13
matrix 32
  addition of 34
  determinant of 34
  inputting 32
  inverse of 34
  multiplication of 34
  powers of 34
  scalar multiple of 34
  subtraction of 34
  transpose of 34
`matrix` 32
Maxima 4
  installing 7
menus, wxMaxima 8
multiplication 13
  matrix 34
  scalar and matrix 34
  scalar and vector 31

natural logarithm 13
not equals, assumption 49, 50
`notequal` 49, 50
`numer` 16
numerical solution of equations 44

output line number 10

`playback` 58
plot style 42
`plot2d` 35
  using command-line interface 60
plotting 35
  axis scales 39
  changing default behaviour 40
  changing line thickness 60
  changing plot style 42
  changing properties 38, 60
  colours 38
  graphs of discrete data 41
  graphs of functions 36
  legend 37
  size of graph, in wxMaxima 40
  using command-line interface 60
  vertical range 36
`points` 42
powers 13
  of a matrix 34
printing
  using the command-line interface 57
  using wxMaxima 19
prompt, wxMaxima 9

rearranging an equation 44
reusing commands
  using the command-line interface 56
`rhs` 43, 44
right-hand side, of an equation 43, 44
roots
  higher 15
  of an equation 43
  square 13

`save` 59
saving your work
  using the command-line interface 57
  using wxMaxima 18
scalar multiple
  of a matrix 34
  of a vector 31
Schelter, William 4
scientific notation 17
screen magnifier, use with wxMaxima 52
screenreader 52
screenshot 19

`sec` 13
`second` 27
series, summing 30
`set_plot_option` 40
simultaneous equations 43
`sin` 13
size of graph, in wxMaxima 40
`solve` 43
solving equations
  analytically 43
  numerically 44
  simultaneously 43
`sqrt` 13
square root 13
subject of equation, changing 44
`subst` 24
substitute 24
  using `ev` 24
  using `subst` 24
subtraction 13
  of matrices 34
  of vectors 31
`sum` 30
summing a list or series 30
symbolic computation 16
syntax 12

table of contents, wxMaxima 9
`tan` 13
toolbar, wxMaxima 9
transcript, creating using the command-line interface 57
transpose
  of a matrix 34
  of a vector 31
`transpose`
  applied to a matrix 34
  applied to a vector 31
trigonometric functions 13
  inverse 13

`values` 23
variable 22
  assign 22
  assumption 49, 50
  define 22
  delete 24
vector 31
  addition of 31
  inputting 31
  scalar multiple of 31
  subtraction of 31
  transpose of 31

worksheet
  wxMaxima 8
`writefile` 58
wxMaxima 6, 8
  abort calculation 17
  cell
    marker 9
  changing colours and fonts 51
  closing 12
  configure 11, 15
  file types 18
  input line number 10
  input prompt 9
  interrupt calculation 17
  line number
    input 10
    output 10
  menus 8
  output line number 10
  printing 19
  prompt, input 9
  saving work 18
  table of contents 9
  toolbar 9
  use with keyboard alone 51
  worksheet 8
`wxplot2d` 35
`wxplot_size` 40